

SIMATIC

TIA Portal Openness: API für die Automatisierung von Engineering- Workflows

Systemhandbuch

Hinweise zur Cybersecurity

1

Liesmich zu
TIA Portal Openness

2

Was ist neu in TIA Portal
Openness?

3

Grundlagen

4

TIA Portal Openness API

5

Export/Import

6

Wesentliche Änderungen

7

Online-Dokumentation

Rechtliche Hinweise

Warnhinweiskonzept

Dieses Handbuch enthält Hinweise, die Sie zu Ihrer persönlichen Sicherheit sowie zur Vermeidung von Sachschäden beachten müssen. Die Hinweise zu Ihrer persönlichen Sicherheit sind durch ein Warndreieck hervorgehoben, Hinweise zu alleinigen Sachschäden stehen ohne Warndreieck. Je nach Gefährdungsstufe werden die Warnhinweise in abnehmender Reihenfolge wie folgt dargestellt.

GEFAHR

bedeutet, dass Tod oder schwere Körpverletzung eintreten wird , wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

WARNUNG

bedeutet, dass Tod oder schwere Körpverletzung eintreten kann , wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

VORSICHT

bedeutet, dass eine leichte Körpverletzung eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

ACHTUNG

bedeutet, dass Sachschaden eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Beim Auftreten mehrerer Gefährdungsstufen wird immer der Warnhinweis zur jeweils höchsten Stufe verwendet. Wenn in einem Warnhinweis mit dem Warndreieck vor Personenschäden gewarnt wird, dann kann im selben Warnhinweis zusätzlich eine Warnung vor Sachschäden angefügt sein.

Qualifiziertes Personal

Das zu dieser Dokumentation zugehörige Produkt/System darf nur von für die jeweilige Aufgabenstellung **qualifiziertem Personal** gehandhabt werden unter Beachtung der für die jeweilige Aufgabenstellung zugehörigen Dokumentation, insbesondere der darin enthaltenen Sicherheits- und Warnhinweise. Qualifiziertes Personal ist auf Grund seiner Ausbildung und Erfahrung befähigt, im Umgang mit diesen Produkten/Systemen Risiken zu erkennen und mögliche Gefährdungen zu vermeiden.

Bestimmungsgemäßer Gebrauch von Siemens-Produkten

Beachten Sie Folgendes:

WARNUNG

Siemens-Produkte dürfen nur für die im Katalog und in der zugehörigen technischen Dokumentation vorgesehenen Einsatzfälle verwendet werden. Falls Fremdprodukte und -komponenten zum Einsatz kommen, müssen diese von Siemens empfohlen bzw. zugelassen sein. Der einwandfreie und sichere Betrieb der Produkte setzt sachgemäßen Transport, sachgemäße Lagerung, Aufstellung, Montage, Installation, Inbetriebnahme, Bedienung und Instandhaltung voraus. Die zulässigen Umgebungsbedingungen müssen eingehalten werden. Hinweise in den zugehörigen Dokumentationen müssen beachtet werden.

Marken

Alle mit dem Schutzrechtsvermerk ® gekennzeichneten Bezeichnungen sind eingetragene Marken der Siemens Aktiengesellschaft. Die übrigen Bezeichnungen in dieser Schrift können Marken sein, deren Benutzung durch Dritte für deren Zwecke die Rechte der Inhaber verletzen kann.

Haftungsausschluss

Wir haben den Inhalt der Druckschrift auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Dennoch können Abweichungen nicht ausgeschlossen werden, so dass wir für die vollständige Übereinstimmung keine Gewähr übernehmen. Die Angaben in dieser Druckschrift werden regelmäßig überprüft, notwendige Korrekturen sind in den nachfolgenden Auflagen enthalten.

Inhaltsverzeichnis

1	Hinweise zur Cybersecurity	21
2	Liesmich zu TIA Portal Openness	23
2.1	Abgekündigte Features in TIA Portal Openness	23
2.2	Liesmich	26
2.3	Größere Änderungen für Langzeitstabilität in TIA Portal Openness V19	30
2.4	Hinweise zum Schreiben von langfristig stabilem Code	32
3	Was ist neu in TIA Portal Openness?	35
4	Grundlagen	39
4.1	Installation	39
4.1.1	Voraussetzungen für TIA Portal Openness	39
4.1.2	TIA Portal Openness installieren	40
4.1.3	Benutzer der Benutzergruppe "Siemens TIA Openness" hinzufügen	41
4.1.4	Verwendung von Windows-Diensten	46
4.1.5	Auf das TIA Portal zugreifen	47
4.1.6	Konfigurationen	48
4.2	Aufgaben von Openness	50
4.2.1	Einführung	50
4.2.2	Einsatzmöglichkeiten	52
4.2.3	Export/Import	53
5	TIA Portal Openness API	55
5.1	TIA Portal Openness-Objekt	55
5.1.1	TIA Portal Openness-Objektmodell	55
5.1.2	Bausteine und Typen des TIA Portal Openness-Objektmodells	61
5.1.3	Hierarchie von Hardware-Objekten des Objektmodells	67
5.1.4	Objektliste	69
5.2	Allgemeine Funktionen	73
5.2.1	Einleitung	73
5.2.2	Informationen über installierte TIA Portal Openness-Versionen	73
5.2.3	Unterstützung von IntelliSense durch TIA Portal Openness	75
5.2.4	Standard-Bibliotheken	76
5.2.5	Einsatz der Codebeispiele	77
5.2.6	Beispielprogramm	79
5.2.7	Programmierschritte	83
5.2.8	Verbindung zum TIA Portal aufbauen	84
5.2.9	TIA Portal Openness-Firewall	90
5.2.10	Event-Handler	95
5.2.11	Dialoge mit Systemmeldungen programmgesteuert bestätigen	97
5.2.12	Verbindung zum TIA Portal beenden	101
5.2.13	Diagnoseschnittstellen im TIA Portal	102
5.2.14	Ausschließlicher Zugriff	107

5.2.15	Dynamisches Verhalten	109
5.2.16	Arbeiten mit Assoziationen	114
5.2.17	Mit Zusammensetzungen arbeiten	115
5.2.18	Objektgleichheit prüfen	116
5.2.19	Lese-Operationen für Attribute	117
5.2.20	Transaktionsbehandlung	118
5.2.21	Ein Objekt DirectoryInfo/FileInfo erstellen	123
5.2.22	Unterstützung der Selbstbeschreibung für Attribute, Navigatoren, Aktionen und Dienste...	126
5.2.23	Attribute von Bausteinen, DBs und UDTs festlegen	129
5.2.24	Anmerkungen zur Leistung von TIA Portal Openness	131
5.3	Funktionen für Projekte und Projektdaten	131
5.3.1	Projekt öffnen	131
5.3.2	Projekt anlegen	136
5.3.3	Auf allgemeine Einstellungen des TIA Portals zugreifen.....	137
5.3.4	Projekt archivieren und abrufen.....	141
5.3.5	Schreibgeschütztes TIA Portal-Projekt öffnen	145
5.3.6	Auf Sprachen zugreifen	146
5.3.7	Objektstruktur und -Attribute ermitteln	148
5.3.8	Auf Software-Ziel zugreifen	150
5.3.9	Auf mehrsprachige Texte zugreifen und sie enumerieren	151
5.3.10	Projekteigenschaften aktualisieren	153
5.3.11	Projektbezogene Attribute lesen	154
5.3.12	Projektgrafik löschen	156
5.3.13	Projekt übersetzen	157
5.3.14	Ein Projekt in UMAC ändern.....	159
5.3.15	Projekt speichern	160
5.3.16	Projekt schließen.....	162
5.3.17	Export/Import von Systemdiagnoseeinstellungen	162
5.4	Funktionen für den Zugriff auf das Teamcenter Gateway.....	164
5.4.1	Verbindung zum Teamcenter Gateway herstellen	164
5.4.2	Verbindung zum Teamcenter Gateway trennen.....	165
5.4.3	Verbindung mit ConnectSSO herstellen	167
5.4.4	Projekt/globale Bibliothek in Teamcenter speichern.....	168
5.4.5	Projekt/globale Bibliothek mit Proxy-Objekt speichern	171
5.4.6	Projekt/globale Bibliothek mit neuem Element speichern	173
5.4.7	Auf benutzerdefinierte Attribute in Teamcenter zugreifen	174
5.4.8	Projekt in Teamcenter als neues Element speichern	178
5.4.9	Projekt als neues Element mit Proxy-Objekt speichern.....	182
5.4.10	Projekt in Teamcenter als neue Version speichern	187
5.4.11	Projekt/globale Bibliothek aus Teamcenter laden.....	191
5.4.12	Projekt/globale Bibliothek in Teamcenter suchen.....	193
5.4.13	Datensatz für Projekt/globale Bibliothek in Teamcenter auschecken.....	195
5.4.14	Datensatz für Projekt/globale Bibliothek in Teamcenter einchecken	197
5.4.15	Checkout für Datensatz in Teamcenter abbuchen.....	199
5.5	Funktionen für Verbindungen	201
5.5.1	Konfigurierbare Attribute einer Port-zu-Port-Verbindung	201
5.6	Funktionen für Bibliotheken	205
5.6.1	Funktionen auf Objekte und Instanzen	205
5.6.2	Auf globale Bibliotheken zugreifen	206
5.6.3	Zugriff auf Bibliotheksbasistypen	208

5.6.4	Zugriff auf Sprachen der globalen Bibliothek	209
5.6.5	Bibliotheken öffnen	211
5.6.6	Bibliotheken vergleichen	213
5.6.7	Typ aus Versionsobjekt erstellen	216
5.6.8	Offene Bibliotheken enumerieren	218
5.6.9	Bibliotheken speichern und schließen	219
5.6.10	Bibliothek archivieren und abrufen	220
5.6.11	Globale Bibliotheken erstellen	223
5.6.12	Auf Ordner in einer Bibliothek zugreifen	224
5.6.13	Auf Typen zugreifen	228
5.6.14	Auf Typ-Versionen zugreifen	230
5.6.15	Zugriff auf Bausteine in Bibliotheken	235
5.6.16	Auf Instanzen zugreifen	237
5.6.17	Auf Kopiervorlagen zugreifen	239
5.6.18	Masterkopie aus einem Projekt in Bibliothek erzeugen	241
5.6.19	Ein Objekt aus einer Masterkopie erstellen	243
5.6.20	Masterkopien kopieren	245
5.6.21	Ermitteln veralteter Typinstanzen	246
5.6.22	Projekt aktualisieren	249
5.6.23	Bibliothek aktualisieren	251
5.6.24	Bibliothek bereinigen	252
5.6.25	Projekt mit Projektbibliothek harmonisieren	253
5.6.26	Struktur während der Bibliotheksaktualisierung aktualisieren	256
5.6.27	Bibliothekstyp aktualisieren	265
5.6.28	Bibliotheksinhalte löschen	267
5.6.29	Standardversion eines Typs festlegen	269
5.6.30	Standardversion für einen Typ abrufen	270
5.6.31	Verbesserte Typkonsistenz	271
5.6.32	PLC-Meldetextlisten und Masterkopien in Bibliotheken verwalten	273
5.7	Funktionen für den Zugriff auf Geräte, Netzwerke und Verbindungen	276
5.7.1	Editor "Geräte & Netze" öffnen	276
5.7.2	PLC-Target und HMI-Target abfragen	277
5.7.3	Auf Attribute eines Adressobjekts zugreifen	279
5.7.4	Aufrufen eines Modulkansals	281
5.8	Funktionen für Netzwerke	283
5.8.1	Subnetz anlegen	283
5.8.2	Auf Subnetze zugreifen	284
5.8.3	Auf interne Subnetze zugreifen	285
5.8.4	Typkennung von Subnetzen abrufen	286
5.8.5	Attribute eines Subnetzes aufrufen	287
5.8.6	Globales Subnetz löschen	294
5.8.7	Alle Beteiligten eines Subnetzes enumerieren	294
5.8.8	IO-Systeme eines Subnetzes enumerieren	295
5.8.9	Auf Knoten zugreifen	295
5.8.10	Attribute eines Knotens aufrufen	296
5.8.11	Knoten mit einem Subnetz verbinden	301
5.8.12	Knoten von einem Subnetz trennen	301
5.8.13	IO-System erstellen	302
5.8.14	Attribute eines IO-Systems aufrufen	303
5.8.15	IO-Connector mit einem IO-System verbinden	303
5.8.16	Mastersystem oder IO-System einer Schnittstelle abrufen	304

5.8.17	IO-Controller abrufen	305
5.8.18	IO-Connector abrufen	306
5.8.19	IO-Connector von einem IO-System oder einem DP-Mastersystem trennen.....	307
5.8.20	Attribute eines DP-Mastersystems aufrufen.....	308
5.8.21	Auf Attribute von PN Driver zugreifen	309
5.8.22	Attribute eines PROFINET IO-Systems aufrufen.....	309
5.8.23	DP-Mastersystem löschen.....	311
5.8.24	Profinet IO-System löschen	312
5.8.25	DP-Mastersystem erstellen	312
5.8.26	Port-Verschaltungsinformationen des Port-Geräteelements aufrufen	313
5.8.27	Attribute der Port-Verschaltung	314
5.8.28	Attribute eines Ports aufrufen	317
5.8.29	DP-Mastersysteme eines Subnetzes enumerieren.....	318
5.8.30	Zugeordnete IO-Connectors enumerieren	319
5.8.31	DP-IO-Connector mit einem DP-Mastersystem verbinden	320
5.8.32	Zugriff auf AS-i-Profil und Parameterattribute für virtuelle Slaves	320
5.8.33	Zugriff auf CM DP als DP-Slave und Transferbereich	323
5.8.34	Kopplung 1516 taktsynchron zentral und dezentral Profinet	324
5.8.35	Openness für CP 1604/CP 1616/CP 1626	327
5.8.36	Openness-Transferbereiche für PN/PN-Koppler	330
5.8.37	Virtuelle Openness-Module/Submodule für ET 200SP PN HF	335
5.8.38	Zugriff auf Domänen-Einstellungen	336
5.8.39	Auf Attribut Redundanzbetrieb zugreifen	339
5.8.40	CCDX-Transferbereich erstellen.....	340
5.8.41	CCDX-Transferbereich löschen	342
5.8.42	Konfiguration von CCDX-Transferbereichen ändern	343
5.9	Funktionen auf Geräte	345
5.9.1	Obligatorische Attribute von Geräten	345
5.9.2	Typkennung von Geräten und Geräteelementen abrufen	347
5.9.3	App-ID in Gerät und Geräteelementen einstellen	349
5.9.4	App-ID in Gerät und Geräteelementen abrufen	350
5.9.5	App-ID in Gerät und Geräteelementen entfernen	351
5.9.6	Erstellen eines Geräts.....	352
5.9.7	Geräte enumerieren.....	353
5.9.8	Auf Geräte zugreifen	356
5.9.9	Auf den TIA Portal-Hardwarekatalog zugreifen.....	358
5.9.10	Löschen eines Geräts	361
5.9.11	Massenänderung von Hardware-Parametern	362
5.10	Funktionen auf Geräteelemente	364
5.10.1	Obligatorische Attribute von Geräteelementen	364
5.10.2	Ein Geräteelement erstellen und stecken	366
5.10.3	Geräteelemente in einen anderen Steckplatz verschieben	369
5.10.4	Geräteelement kopieren.....	370
5.10.5	Einstellungen für benutzerdefiniertes Logo vornehmen	371
5.10.6	Geräteelement löschen	373
5.10.7	Geräteelemente enumerieren	373
5.10.8	Geräteelemente aufrufen	375
5.10.9	Auf das Geräteelement als Schnittstelle zugreifen.....	378
5.10.10	Auf "Gerät ändern" zugreifen.....	379
5.10.11	Auf Attribute einer I/O-Geräteschnittstelle zugreifen	380
5.10.12	Subnetz eines Geräteelements abrufen	382

5.10.13	Auf Attribute des IoController zugreifen.....	383
5.10.14	Auf Attribute des IoConnector zugreifen.....	384
5.10.15	Auf einen Adresscontroller zugreifen.....	387
5.10.16	Auf Adressen zugreifen.....	388
5.10.17	Auf "Hardwarekennung" zugreifen.....	390
5.10.18	Auf Hardwarekennungscontroller zugreifen.....	390
5.10.19	Auf Kanäle von Geräteelementen zugreifen.....	391
5.10.20	Auf normierte Typkennungen zugreifen.....	393
5.10.21	PSC-Datei erstellen und exportieren.....	393
5.10.22	Verwalten von Verbindungen bei Erweiterungsbaugruppenträgern.....	394
5.11	Funktionen für den Zugriff auf die Daten eines PLC-Geräts.....	395
5.11.1	Funktionen zum Laden von Daten ins PLC-Gerät.....	395
5.11.1.1	Zugriff auf Systemdiagnose-Eigenschaften von Plus-PLCs.....	395
5.11.1.2	PC-System laden.....	397
5.11.1.3	Herunterladen zu den PLCs.....	400
5.11.1.4	Herunterladen der PLC in einen Windows-Ordner.....	412
5.11.1.5	Master Secret in eine PLC laden.....	415
5.11.1.6	Master Secrets in PLCs verwalten.....	417
5.11.1.7	Ein PLC Master Secret in einer PLC setzen / löschen.....	420
5.11.1.8	PLC starten und stoppen.....	421
5.11.1.9	Unterstützung von Callbacks.....	422
5.11.1.10	PLC durch Passwort schützen.....	424
5.11.1.11	Handhabung bausteingebundener PLC-Passwörter.....	425
5.11.1.12	Hochladen der PLC.....	426
5.11.1.13	PLC-Software vergleichen.....	434
5.11.1.14	PLC-Programm aktualisieren.....	437
5.11.1.15	PLC-Hardware vergleichen.....	438
5.11.1.16	Passwortrichtlinie für PLC festlegen.....	439
5.11.1.17	Verwalten von dynamischen Zertifikatseinstellungen.....	441
5.11.2	Funktionen für den Zugriff auf den PLC-Dienst.....	445
5.11.2.1	Einstellung der Zugriffsstufen.....	445
5.11.2.2	Auf Software-Prüfsumme zugreifen.....	447
5.11.2.3	PC-Schnittstelle zuordnen.....	448
5.11.2.4	Status einer PLC ermitteln.....	450
5.11.2.5	Auf Parameter einer Online-Verbindung zugreifen.....	451
5.11.2.6	Zugriff auf den Fingerabdruck zum schnellen Vergleich von Stationen.....	455
5.11.2.7	Unter STEP 7 auf Cross Reference Service zugreifen.....	463
5.11.2.8	Querverweise für STEP 7 abrufen.....	464
5.11.2.9	PLC von R/H-System online setzen.....	469
5.11.2.10	Auf Softwarebehälter über primären PLC eines R/H-Systems zugreifen.....	472
5.11.2.11	PLCs eines R/H-Systems laden.....	473
5.11.2.12	Online-Verbindung zur PLC aufbauen oder trennen.....	479
5.11.2.13	Projektsprache der PLC zuordnen.....	480
5.11.2.14	Beobachtungs- und Forcetabellen für Webserver- und PLC-Display zuordnen.....	481
5.11.2.15	Zertifikat verwalten.....	484
5.11.2.16	Webserver-Benutzer von SIWAREX-Modulen verwalten.....	488
5.11.2.17	Verwalten der Syslog-Konfiguration.....	490
5.11.2.18	Unterstützung der sicheren S7-Kommunikation mit TLS.....	492
5.11.2.19	Modulbeschreibung aktualisieren.....	497
5.11.2.20	Unterstützung von UMAC auf PLC-Gerät.....	498
5.11.2.21	Zugänglichkeit.....	502
5.11.2.22	Zugreifen auf den Konfigurationsanbieter für die PLC-Zugriffskontrolle.....	503

5.11.2.23	Display-Passwort festlegen	505
5.11.2.24	Zugriff auf Webserver und OPC UA-Benutzerverwaltung	506
5.11.3	Bausteine	509
5.11.3.1	Gruppe "Programmbausteine" abfragen.....	509
5.11.3.2	Systemgruppe für Systembausteine abfragen	510
5.11.3.3	Systemuntergruppen enumerieren	510
5.11.3.4	Benutzerdefinierte Bausteingruppen enumerieren	512
5.11.3.5	Alle Bausteine enumerieren	513
5.11.3.6	Informationen eines Bausteins/Anwenderdatentyps abfragen	514
5.11.3.7	Einen Baustein schützen und Schutz aufheben	515
5.11.3.8	Baustein löschen.....	518
5.11.3.9	Gruppe für Bausteine erzeugen	519
5.11.3.10	Gruppe für Bausteine löschen.....	519
5.11.3.11	Auf Attribute sämtlicher Bausteine zugreifen	520
5.11.3.12	Einen ProDiag-FB anlegen	522
5.11.3.13	Auf Überwachungen und Eigenschaften des ProDiag-FB zugreifen	523
5.11.3.14	ProDiag-FB für DBs & Variablentabelle zuweisen	524
5.11.3.15	ProDiag-FB-Bausteine und -Attribute lesen.....	526
5.11.3.16	ProDiag-Meldungen exportieren.....	527
5.11.3.17	Externe Datei hinzufügen.....	528
5.11.3.18	Quelle aus Baustein generieren	529
5.11.3.19	Bausteine aus Quelle erzeugen.....	530
5.11.3.20	Generierung aus einer Quelle mit bekanntem Quellformat	531
5.11.3.21	Anwenderdatentyp löschen.....	534
5.11.3.22	Auf Datentypen des PLC-Systems zugreifen	534
5.11.3.23	Externe Datei löschen	535
5.11.3.24	Bausteeditor starten.....	536
5.11.3.25	Bausteine mit Hilfe von Fingerabdrücken ändern	537
5.11.3.26	Bausteine für benutzerdefinierte Seiten generieren/löschen	538
5.11.3.27	Schreibzugriff für Prioritätsattribut des OB-Bausteins.....	541
5.11.3.28	Bausteine/UDTs aus einer externen Quelldatei in einer bestimmten Benutzergruppe erzeugen	541
5.11.3.29	Ladbare Datei für Bausteine außerhalb von Software Units erzeugen	543
5.11.4	Technologieobjekte.....	546
5.11.4.1	Übersicht über Funktionen für Technologieobjekte	546
5.11.4.2	Übersicht über Technologieobjekte und Versionen.....	547
5.11.4.3	Überblick der Datentypen.....	549
5.11.4.4	Zusammensetzung von Technologieobjekten abfragen.....	550
5.11.4.5	Technologieobjekt erstellen	550
5.11.4.6	Technologieobjekt löschen.....	551
5.11.4.7	Gruppe für Technologieobjekte erstellen	552
5.11.4.8	Gruppe für Technologieobjekte löschen.....	553
5.11.4.9	Technologieobjekt übersetzen.....	553
5.11.4.10	Technologieobjekte enumerieren	555
5.11.4.11	Gruppen für Technologieobjekte enumerieren.....	555
5.11.4.12	Technologieobjekt finden	557
5.11.4.13	Parameter eines Technologieobjekts enumerieren	557
5.11.4.14	Parameter eines Technologieobjekts finden	558
5.11.4.15	Parameter eines Technologieobjekts lesen.....	559
5.11.4.16	Parameter eines Technologieobjekts schreiben	560
5.11.4.17	S7-1200 Motion Control.....	561
5.11.4.18	S7-1500 Motion Control.....	569

5.11.4.19	PID-Regelung	599
5.11.4.20	Zählen	599
5.11.4.21	Easy Motion Control.....	600
5.11.5	Variablen und Variablentabellen.....	600
5.11.5.1	Starten des PLC-Variableneditors	600
5.11.5.2	Systemgruppen für PLC-Variablen abfragen	601
5.11.5.3	PLC-Variablentabelle anlegen	602
5.11.5.4	Benutzerdefinierte Gruppen für PLC-Variablen enumerieren	602
5.11.5.5	Benutzerdefinierte Gruppen für PLC-Variablen erzeugen	603
5.11.5.6	Benutzerdefinierte Gruppen für PLC-Variablen löschen	604
5.11.5.7	PLC-Variablentabellen in einem Ordner enumerieren	605
5.11.5.8	Informationen einer PLC-Variablentabelle abfragen	606
5.11.5.9	Zeitpunkt der letzten Änderung einer PLC-Variablentabelle lesen	607
5.11.5.10	PLC-Variablentabelle aus einer Gruppe löschen	608
5.11.5.11	PLC-Variablen enumerieren	609
5.11.5.12	Auf PLC-Variablen zugreifen	609
5.11.5.13	Auf PLC-Konstanten zugreifen	611
5.11.6	Funktionen für Software-Units	613
5.11.6.1	Auf Software-Units zugreifen	613
5.11.6.2	Mit Software-Units arbeiten	617
5.11.6.3	Auf Software-Units zugrundeliegende Objekte zugreifen	621
5.11.6.4	Auf vorhandene Beziehungen einer Unit zugreifen	624
5.11.6.5	Eigenschaften von Software-Units aktualisieren.....	626
5.11.6.6	Objekt einer Software-Unit veröffentlichen	629
5.11.6.7	Externe Quellen in Units hinzufügen	631
5.11.6.8	Units als Masterkopien	634
5.11.6.9	Vorhandene Beziehungen aktualisieren und Beziehungen erstellen/löschen	636
5.11.6.10	Zugriff auf Namespaces für Software-Einheiten	642
5.11.6.11	Zugriff auf Namespaces für Programmbausteine.....	643
5.11.6.12	Auf Namespaces für UDT zugreifen.....	644
5.11.6.13	Ladbare Datei für Software Units erzeugen	647
5.11.7	Funktionen für fehlersichere Units.....	650
5.11.7.1	Fehlersichere Unit generieren	650
5.11.7.2	Auf die Safety Unit zugreifen	653
5.11.7.3	Beziehungen für die Safety Unit anlegen/löschen.....	655
5.11.7.4	Elemente/Objekte einer Safety Unit anlegen bzw. darauf zugreifen	657
5.11.7.5	Überwachungen unter einer Safety Unit exportieren/importieren.....	659
5.11.7.6	Bausteine unterhalb einer Safety Unit veröffentlichen	662
5.12	Funktionen für den Zugriff auf die Daten eines HMI-Gerätes.....	664
5.12.1	Bilder.....	664
5.12.1.1	Benutzerdefinierte Bilderordner erzeugen.....	664
5.12.1.2	Bild aus einem Ordner löschen	664
5.12.1.3	Bildvorlage aus einem Ordner löschen	665
5.12.1.4	Alle Bilder aus einem Ordner löschen	666
5.12.2	Zyklen	667
5.12.2.1	Zyklus löschen	667
5.12.3	Textlisten	667
5.12.3.1	Textliste löschen	667
5.12.4	Grafiklisten	668
5.12.4.1	Grafikliste löschen	668
5.12.5	Verbindungen.....	669
5.12.5.1	Verbindung löschen	669

5.12.6	Variablentabelle.....	669
5.12.6.1	Benutzerdefinierte Ordner für HMI-Variablen erzeugen	669
5.12.6.2	Variablen einer HMI-Variablentabelle enumerieren	670
5.12.6.3	Einzelne Variable aus einer HMI-Variablentabelle löschen	671
5.12.6.4	Variablentabelle aus einem Ordner löschen	671
5.12.7	VB-Skripte	672
5.12.7.1	Benutzerdefinierte Ordner für Skripte erzeugen	672
5.12.7.2	VB-Skript aus einem Ordner löschen	673
5.12.7.3	Benutzerdefinierten Ordner eines Bediengeräts löschen	673
5.13	Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified).....	674
5.13.1	HMI-Software (RT Unified)	674
5.13.1.1	HMI-Softwareobjekt (RT Unified)	674
5.13.1.2	Beschreibung HMI-Software (RT Unified)	676
5.13.1.3	AlarmClasses (RT Unified)	678
5.13.1.4	AlarmLogs (RT Unified).....	683
5.13.1.5	AnalogAlarms (RT Unified).....	687
5.13.1.6	AuditTrails (RT Unified)	693
5.13.1.7	Verbindungen (RT Unified)	696
5.13.1.8	DataLogs (RT Unified).....	701
5.13.1.9	DiscreteAlarms (RT Unified)	706
5.13.1.10	LoggingTags (RT Unified).....	711
5.13.1.11	RuntimeSettings (RT Unified).....	716
5.13.1.12	SystemTags (RT Unified)	719
5.13.1.13	Bilder (RT Unified)	721
5.13.1.14	Bildgruppen (RT Unified)	838
5.13.1.15	Variablen (RT Unified)	840
5.13.1.16	TagTables (RT Unified)	852
5.13.1.17	Variablentabellengruppen (RT Unified)	862
5.13.2	Auf weitere Objekte zugreifen (RT Unified)	864
5.13.2.1	Umbenennen von Eigenschaften und Datentypen (RT Unified).....	864
5.13.2.2	Fehler abfragen (RT Unified).....	866
5.13.2.3	Auf einfache Dynamisierung zugreifen (RT Unified).....	869
5.13.2.4	Auf den Gerätenamen eines HMI-Panels zugreifen (RT Unified)	873
5.13.2.5	Zugriff auf Hierarchien im Common Plant Model (RT Unified)	874
5.13.2.6	Zugriff auf Instanzen im Common Plant Model (RT Unified).....	880
5.13.2.7	Auf Eigenschaften für Schnittstellen/Archivvariablen von Anlagenobjektinstanzen zugreifen (RT Unified)	884
5.13.2.8	Lizenz für den Zugriff auf ein Unified-Gerät prüfen (RT Unified)	890
5.13.2.9	Querverweise für WinCC Unified abrufen (RT Unified)	893
5.13.2.10	Mit Dynamisierung und Ereignissen für Bilder/Bildobjekte über Skripte arbeiten (RT Unified)	896
5.13.2.11	Mit Dynamisierung für Bilder/Bildobjekte arbeiten (RT Unified).....	899
5.14	Funktionen der Versionskontrollschnittstelle (VCI)	903
5.14.1	Auf die VCI-Systemgruppe im Projekt zugreifen.....	903
5.14.2	Benutzergruppen in VCI-Gruppe enumerieren.....	903
5.14.3	VCI-Benutzergruppe in VCI-Gruppe erstellen	904
5.14.4	Eigenschaften von VCI-Gruppen aktualisieren	905
5.14.5	VCI-Benutzergruppe löschen	906
5.14.6	VCI-Workspaces in VCI-Gruppe enumerieren	906
5.14.7	VCI-Workspace in VCI-Gruppe erstellen	907
5.14.8	VCI-Workspace-Eigenschaften aktualisieren	908

5.14.9	VCI-Workspace löschen	909
5.14.10	VCI-Workspace-Zuordnungen in VCI enumerieren	910
5.14.11	VCI-Workspace-Zuordnung in VCI-Workspace erstellen	911
5.14.12	VCI-Workspace-Zuordnungen aktualisieren	911
5.14.13	VCI-Workspace-Zuordnung löschen	912
5.14.14	Workspace-Zuordnung synchronisieren	913
5.14.15	Auf den Synchronisationsstatus untergeordneter Objekte zugreifen	916
5.14.16	VCI-Workspace mit Workspace-Sprache erstellen	917
5.14.17	Initialisierung des VCI-Objektstatus	919
5.14.18	Importieren von Simatic ML mit inaktiver Kultur	920
5.15	Funktionsunterstützung für Mehrbenutzerbetrieb	921
5.15.1	Verbindung zum Projektserver erstellen	921
5.15.2	Projektserververbindung bearbeiten	922
5.15.3	Projektserververbindung löschen	923
5.15.4	Herstellen der Verbindung zum Projektserver aus den TIA Portal-Einstellungen heraus	924
5.15.5	Ein Projekt im Server hinzufügen	925
5.15.6	Serverprojekte abrufen	926
5.15.7	Verfügbare lokale Sitzungen abrufen	928
5.15.8	Eine lokale Sitzung erstellen	928
5.15.9	Eine lokale Sitzung speichern	929
5.15.10	Lokale Sitzung auf dem Server löschen	930
5.15.11	Lokale/exklusive Sitzung öffnen	931
5.15.12	Serverprojekt öffnen	932
5.15.13	Markierte Objekte prüfen	933
5.15.14	Eine lokale Sitzung schließen	935
5.15.15	Änderung des Serverprojekts übernehmen	936
5.15.16	LockStateProvider abrufen	937
5.15.17	Gesperrtes Projekt prüfen	938
5.15.18	Eigentümer der Sperre abrufen	938
5.15.19	Aktualität der Sitzung prüfen	939
5.15.20	Markierungen aus lokaler Sitzung abrufen	940
5.16	Funktionen für den Zugriff auf Daten der Benutzerverwaltung	942
5.16.1	Funktionen für den Zugriff auf Benutzer, Rollen und Funktionsrechte	942
5.16.1.1	Projekt schützen	942
5.16.1.2	Engineering-Funktionsrechte	943
5.16.1.3	Funktionsrechte für Geräte	945
5.16.1.4	Systemrollen	946
5.16.1.5	Benutzerdefinierte Rollen	948
5.16.1.6	Benutzer eines Projekts	950
5.16.2	Funktionen für globale UMAC-Benutzer und UMC-Server	952
5.16.2.1	Authentifizierung für die Verbindung mit einem UMC-Server	952
5.16.2.2	Eine UMC-Benutzergruppe von einem UMC-Server abrufen	953
5.16.2.3	Einen UMC-Benutzer von einem UMC-Server abrufen	954
5.16.2.4	Eine UMC-Benutzergruppe hinzufügen	955
5.16.2.5	Alle UMC-Gruppen in TIA Portal abrufen	955
5.16.2.6	Einen UMC-Benutzer in TIA Portal hinzufügen	956
5.16.2.7	Alle UMC-Benutzer in einem TIA Portal-Projekt abrufen	957
5.16.2.8	Hinzugefügten UMC-Benutzer in TIA Portal suchen	958
5.16.2.9	Hinzugefügte UMC-Benutzergruppe in einem TIA Portal-Projekt suchen	958
5.16.2.10	Einem UMC-Benutzer eine Rolle zuweisen	959
5.16.2.11	Liste zugewiesener Rollen für UMC-Benutzer und UMC-Benutzergruppen abrufen	960

5.16.2.12	Einer UMC-Benutzergruppe eine Rolle zuweisen	960
5.16.2.13	Einem UMC-Benutzer und einer UMC-Benutzergruppe eine Rolle entziehen	961
5.16.2.14	UMC-Benutzer und UMC-Benutzergruppe aktivieren/deaktivieren.....	962
5.16.2.15	Status eines UMC-Benutzers bzw. einer UMC-Benutzergruppe prüfen.....	963
5.16.2.16	UMC-Benutzer und UMC-Benutzergruppe löschen	963
5.16.2.17	UMC-Benutzer synchronisieren.....	964
5.16.2.18	Konsistenz der UMC-Daten überprüfen	965
5.16.2.19	Runtime-Sitzungs-Timeout festlegen	965
5.16.2.20	Passwortrichtlinien für UMAC festlegen	966
5.16.2.21	In geschütztem Projekt auf Anonymous User zugreifen	971
5.16.2.22	Vollständiger Beispielcode.....	972
5.17	Funktionen auf OPC	976
5.17.1	Konfiguration des sicheren Kommunikationsprotokolls für den OPC UA-Server	976
5.17.2	Sicherheitsrichtlinie für OPC UA festlegen.....	978
5.17.3	Zugriff auf den OPC UA-Referenz-Namespace	979
5.17.4	Zugriff auf die OPC UA Server-(SIMATIC)Schnittstelle.....	984
5.17.5	Zugriff auf die OPC UA Server-Schnittstelle	987
5.18	SiVArc Openness	989
5.18.1	Einleitung	989
5.18.2	Openness Lizenz-Handhabung	990
5.18.3	SiVArc-Instanzeigenschaften	990
5.18.4	Kopieren von Regeln oder Gruppen aus der Bibliothek	991
5.18.5	Suchen von Bildregelgruppen und Bildregeln.....	993
5.18.6	Zuweisen von Eigenschaften für Regeln und Regelgruppen	996
5.18.7	Erstellen von Regelobjekt und Regelgruppe	997
5.18.8	Arbeiten mit Bildregeltabellen und -ordnern	998
5.18.8.1	Bildregeltabelle erstellen	998
5.18.8.2	Bildregelordner erstellen	999
5.18.8.3	Zugreifen auf Bildregeln und -gruppen	999
5.18.8.4	Löschen der Bildregeltabelle und des benutzerdefinierten Regelordners	1001
5.18.8.5	Auf die Regeltabelle in Library Type zugreifen	1001
5.18.8.6	Instanz einer Regeltabelle erstellen	1002
5.18.9	Auf Ordner unter "Kopiervorlage/Typ eines Bildes" zugreifen	1002
5.18.10	Zugreifen auf Bibliotheksobjekte	1003
5.18.11	Projektieren des PLC- und HMI-Geräts	1007
5.18.12	Löschen von Regeln und Regelgruppen	1010
5.18.13	Projektieren von Variablenregeln.....	1011
5.18.14	Variablen-/Textdefinitionen konfigurieren	1012
5.18.15	UMAC-Einrichtung für Openness	1014
5.18.16	SiVArc-Generierung.....	1015
5.19	Unterstützung für Openness	1019
5.19.1	Einleitung	1019
5.19.2	Funktionen für Styleguide	1024
5.19.2.1	Zugriff auf Styleguide-Systemordner.....	1024
5.19.2.2	Zugriff auf die Regelsatz-Zusammensetzung	1024
5.19.2.3	Zugriff auf Regelsatz	1025
5.19.2.4	Zugriff auf den Namen des Regelsatzes.....	1025
5.19.2.5	Regelsatz einblenden	1026
5.19.2.6	Regelsatz löschen	1027
5.19.2.7	Den Geltungsbereich eines Regelsatzes modifizieren.....	1028
5.19.2.8	Ausführung von Regelsätzen.....	1033

5.19.2.9	Import und Export von Regelsätzen	1036
5.19.2.10	Arbeiten mit Bibliotheken	1038
5.19.3	Funktionen für den Anwendungstest	1042
5.19.3.1	Zugriff auf den Systemorder für den Anwendungstest	1042
5.19.3.2	Zugriff auf TestCaseComposition	1042
5.19.3.3	Zugriff auf den Testfall	1043
5.19.3.4	Zugriff auf den Namen des Testfalls	1043
5.19.3.5	Zugriff auf den Scope des Testfalls	1044
5.19.3.6	Testfall einblenden	1045
5.19.3.7	Testfall löschen	1046
5.19.3.8	Den Geltungsbereich eines Testfalls modifizieren	1047
5.19.3.9	Ausführung von Testfällen	1049
5.19.3.10	Import/Export von Testfällen mit Openness APIs	1056
5.19.3.11	Mit Bibliotheken arbeiten	1058
5.19.4	Funktionen für Systemtest	1062
5.19.4.1	Zugriff auf den Systemorder für den Systemtest	1062
5.19.4.2	Zugriff auf TestCaseComposition	1062
5.19.4.3	Zugriff auf Systemtestfall	1063
5.19.4.4	Zugriff auf Namen des Systemtestfalls	1063
5.19.4.5	Zugriff auf den Bereich eines Systemtestfalls	1064
5.19.4.6	Bereich eines Systemtestfalls ändern	1065
5.19.4.7	Ausführung von Systemtestfällen	1068
5.19.4.8	Import/Export von Systemtestfällen mit Openness APIs	1071
5.19.4.9	Arbeiten mit Bibliotheken	1073
5.19.5	Exceptions	1077
5.20	Funktionen für SINUMERIK 840D sl	1077
5.20.1	Einführung	1077
5.20.2	Typkennung- Kennzeichnung der Komponenten	1078
5.20.3	Grundlagen	1078
5.20.4	Objektmodell	1081
5.20.5	Codebeispiel	1083
5.20.5.1	Allgemein	1083
5.20.5.2	Erste Schritte in SINUMERIK durchführen	1083
5.20.5.3	NCU anlegen	1083
5.20.5.4	NX-Baugruppe anlegen	1084
5.20.5.5	NX-Baugruppe mit NCU verbinden	1084
5.20.5.6	Safety Integrated aktivieren	1085
5.20.5.7	Auf PLC-Software zugreifen	1086
5.21	Funktionen für SINUMERIK ONE	1087
5.21.1	Einführung	1087
5.21.2	Typkennung-Kennzeichnung der Komponenten	1088
5.21.3	Grundlagen	1088
5.21.4	Objektmodell	1091
5.21.5	Referenz	1093
5.21.5.1	Namespace Siemens.Engineering.MC.DriveConfiguration	1093
5.21.5.2	Namespace Siemens.Engineering.MC.Drives	1102
5.21.5.3	ArchiveProvider	1110
5.21.6	Codebeispiel	1111
5.21.6.1	Allgemein	1111
5.21.6.2	Erste Schritte in SINUMERIK durchführen	1111
5.21.6.3	NCU anlegen	1111

5.21.6.4	NX-Baugruppe anlegen	1111
5.21.6.5	NX-Baugruppe mit NCU verbinden.....	1112
5.21.6.6	Auf NCK-Ereignisse zugreifen	1113
5.21.6.7	Archive erstellen	1114
5.21.6.8	Safety Integrated aktivieren.....	1116
5.21.6.9	Auf PLC-Software zugreifen	1117
5.21.6.10	SINUMERIK PLC-Alarmtexte importieren	1118
5.21.6.11	Beispiele für Namespace Siemens.Engineering.MC.DriveConfiguration.....	1119
5.21.6.12	Beispiele für Namespace Siemens.Engineering.MC.Drives	1122
5.22	Funktionen für SINUMERIK MC	1126
5.22.1	Einführung	1126
5.22.2	Codebeispiel	1127
5.22.2.1	Auf PLC-Software zugreifen	1127
5.23	Funktionen für Startdrive	1128
5.23.1	Einführung	1128
5.23.2	Security-Hinweise	1129
5.23.2.1	Verschlüsselte Kommunikation mit SecureString-Passwörtern	1129
5.23.2.2	Passwörter in TIA Openness	1129
5.23.3	Typidentifizier - Kennzeichnung der Komponenten	1130
5.23.4	Referenzen	1131
5.23.4.1	AddressComposition	1131
5.23.4.2	AddressContext.....	1132
5.23.4.3	AddressloType.....	1132
5.23.4.4	Commissioning	1133
5.23.4.5	ConfigurationEntry	1133
5.23.4.6	DriveDataEncryption (ab SINAMICS FW V6.1).....	1133
5.23.4.7	DriveDomainFunctions.....	1134
5.23.4.8	DriveObject.....	1135
5.23.4.9	DriveObjectActivation	1135
5.23.4.10	DriveObjectContainer.....	1136
5.23.4.11	ModuleAccessPoint	1136
5.23.4.12	DriveObjectTypeHandler.....	1136
5.23.4.13	DriveParameter	1137
5.23.4.14	DriveParameterComposition.....	1139
5.23.4.15	HardwareProjection	1139
5.23.4.16	EncoderConfiguration	1141
5.23.4.17	MotorConfiguration	1142
5.23.4.18	OnlineDriveObject	1143
5.23.4.19	OnlineDriveObjectContainer.....	1143
5.23.4.20	StartDriveDownloadCheckConfiguration	1144
5.23.4.21	SafetyTelegram	1144
5.23.4.22	TechnologyExtension	1145
5.23.4.23	TechnologyExtensionComposition.....	1146
5.23.4.24	TechnologyExtensionInstallationProvider.....	1146
5.23.4.25	TechnologyExtensionPackage.....	1147
5.23.4.26	Telegram	1148
5.23.4.27	TelegramComposition	1149
5.23.4.28	TelegramType	1150
5.23.4.29	TorqueTelegram	1151
5.23.4.30	AxisEncoderHardwareConnectionInterface	1151
5.23.4.31	UmacConfiguration (ab SINAMICS FW V6.1)	1152

5.23.4.32	UpdateChecksums (ab SINAMICS FW V6.1)	1152
5.23.5	Codebeispiele	1153
5.23.5.1	Aktivierungsstatus ermitteln	1153
5.23.5.2	Antriebsfunktionen durchführen	1153
5.23.5.3	Antrieb anlegen	1154
5.23.5.4	Antriebskomponente anlegen	1155
5.23.5.5	Antriebsobjekt ermitteln	1155
5.23.5.6	Antriebsobjekt-Typ ermitteln	1156
5.23.5.7	BICO-Parameter lesen und schreiben	1156
5.23.5.8	Download	1157
5.23.5.9	DRIVE-CLiQ-Verbindungen bearbeiten	1161
5.23.5.10	Erste Schritte in Startdrive durchführen	1162
5.23.5.11	Geber-Typ festlegen	1163
5.23.5.12	Geräte-Projektierung durchführen	1173
5.23.5.13	Hardware Identifier ermitteln	1173
5.23.5.14	Komponente für eine Antriebskomponente anlegen (nur S120)	1174
5.23.5.15	Motor-Typ und Motor-Konfiguration festlegen	1176
5.23.5.16	Parameter lesen und schreiben	1182
5.23.5.17	Parameter online lesen und schreiben	1184
5.23.5.18	Parametrierung speichern	1184
5.23.5.19	SecureString-Passwort vergeben	1185
5.23.5.20	Safety Integrated (Prüfsummen) konfigurieren (ab SINAMICS FW V6.1)	1186
5.23.5.21	Safety Integrated-Telegramme verwenden	1187
5.23.5.22	SIMOGEAR-Artikelnummer für G115D-Antriebe setzen	1188
5.23.5.23	Technology Extensions verwenden	1188
5.23.5.24	Technologieobjekte über Telegrammobjekte mit Hardware-Modulen verbinden	1191
5.23.5.25	Telegramme einfügen und erweitern	1192
5.23.5.26	Torque-Telegramme verwenden	1193
5.23.5.27	UMAC für Antriebe aktivieren/deaktivieren	1194
5.23.5.28	Verschlüsselung der Antriebsdaten/DDE aktivieren (ab SINAMICS FW V6.1)	1194
5.23.5.29	Werkseinstellungen wiederherstellen	1195
5.24	Funktionen für SIMATIC Drive Controller	1196
5.24.1	Zugreifen auf PLC und Kanaleigenschaften von X142	1196
5.24.2	Zugreifen auf die integrierten Eigenschaften von PROFIdrive	1199
5.24.3	Erstellen von SINAMICS Integrated	1201
5.24.4	Abrufen der Adresse des zentralen Bausteins (X142)	1201
5.24.5	Abrufen der Adresse des Antriebsregelungstelegramms	1202
5.25	Funktionen für DCC	1202
5.25.1	Einführung	1202
5.25.2	DCC Openness	1203
5.25.3	Objektmodell DCC Openness	1204
5.25.4	Referenzen	1204
5.25.4.1	DccBlock	1204
5.25.4.2	DccBlockComposition	1205
5.25.4.3	DcbLibrary	1205
5.25.4.4	DcbBlockType	1206
5.25.4.5	DcbLibraryComposition	1206
5.25.4.6	DccConnection	1206
5.25.4.7	DccImportOptions	1207
5.25.4.8	DccChartPartitionComposition	1207
5.25.4.9	DccImportResultData	1208

5.25.4.10	DccParameter	1208
5.25.4.11	DccPin	1209
5.25.4.12	DccPinComposition	1210
5.25.4.13	DriveControlChart	1210
5.25.4.14	DriveControlChartContainer	1212
5.25.4.15	DriveControlChartComposition	1212
5.25.4.16	Import DCB extension library.....	1213
5.25.4.17	Statement.....	1214
5.25.5	Codebeispiele	1215
5.25.5.1	Allgemein	1215
5.25.5.2	Ablaufreihenfolge ändern	1215
5.25.5.3	Ablaufreihenfolge optimieren	1215
5.25.5.4	Auf Pläne zugreifen	1216
5.25.5.5	Alle Pläne exportieren	1216
5.25.5.6	DCB Extension-Bibliothek importieren	1216
5.25.5.7	DCB-Bibliothek Info auslesen	1217
5.25.5.8	DCC-Blöcke bearbeiten	1217
5.25.5.9	DCB-Blocktyp Info auslesen	1218
5.25.5.10	DCC-Parameter bearbeiten	1219
5.25.5.11	Importbericht anzeigen.....	1219
5.25.5.12	Pins bearbeiten	1219
5.25.5.13	Pläne in der Ablaufreihenfolge abrufen	1220
5.25.5.14	Pläne oder Blöcke anhand des Namens finden	1220
5.25.5.15	Plan oder Sub-Plan erstellen	1221
5.25.5.16	Pläne bearbeiten	1221
5.25.5.17	Pläne löschen	1222
5.25.5.18	Pläne abrufen	1222
5.25.5.19	Pläne importieren	1223
5.25.5.20	Pläne exportieren.....	1223
5.25.5.21	Verbindungen bearbeiten	1223
5.25.5.22	Zugriff auf DriveControlChartContainer über DriveObject.....	1224
5.25.6	Ausnahmen DCC Openness	1224
5.25.6.1	Umgang mit Exceptions	1224
5.25.6.2	DccBlockComposition Exceptions	1226
5.25.6.3	DriveControlChart Exceptions.....	1227
5.25.6.4	DriveControlChartComposition Exceptions.....	1227
5.25.6.5	ImportDcbLibrary Exceptions.....	1228
5.26	Ausnahmen	1229
5.26.1	Umgang mit Exceptions	1229
5.27	F-spezifisches Openness	1232
5.27.1	F-spezifisches Openness	1232
5.27.2	SafetyModificationsPossible.....	1233
5.27.3	UsernameForFChangeHistory.....	1234
5.27.4	SafetySignatureProvider	1234
5.27.5	SafetyAdministration	1235
5.27.5.1	SafetyAdministration	1235
5.27.5.2	IsSafetyOfflineProgramPasswordSet	1235
5.27.5.3	SetSafetyOfflineProgramPassword	1236
5.27.5.4	RevokeSafetyOfflineProgramPassword	1236
5.27.5.5	IsLoggedInToSafetyOfflineProgram	1237
5.27.5.6	LoginToSafetyOfflineProgram.....	1237

5.27.5.7	LogoffFromSafetyOfflineProgram	1238
5.27.5.8	Ablaufgruppen.....	1239
5.27.5.9	Sicherheitseinstellungen	1244
5.27.6	SafetyPrintout	1247
6	Export/Import	1249
6.1	Überblick	1249
6.1.1	Grundlagen zum Import/Export	1249
6.1.2	Einsatzgebiet von Import/Export.....	1252
6.1.3	Versionsspezifischer SimaticML-Import	1252
6.1.4	Bearbeiten der XML-Datei.....	1253
6.1.5	Export von Projektierungsdaten	1253
6.1.6	Import von Projektierungsdaten	1255
6.2	Import/Export von Projektdaten.....	1257
6.2.1	Grafiksammlung	1257
6.2.1.1	Export/Import von Grafiken	1257
6.2.1.2	Grafiken eines Projektes exportieren	1258
6.2.1.3	Grafiken in ein Projekt importieren	1259
6.2.2	Projekttexte	1260
6.2.2.1	Export von Projekttexten	1260
6.2.2.2	Import von Projekttexten.....	1261
6.3	Import/Export von Daten eines HMI-Geräts	1263
6.3.1	Datenstruktur für Import/Export	1263
6.3.1.1	Aufbau einer XML-Datei	1263
6.3.1.2	Struktur der Daten für Import/Export	1265
6.3.1.3	Zyklen	1269
6.3.2	Variablentabellen.....	1271
6.3.2.1	HMI-Variablentabellen exportieren	1271
6.3.2.2	HMI-Variablentabelle importieren.....	1274
6.3.2.3	Einzelne Variable einer HMI-Variablentabelle exportieren.....	1275
6.3.2.4	Einzelne Variable in eine HMI-Variablentabelle importieren.....	1276
6.3.2.5	Besonderheiten beim Export/Import von HMI-Variablen	1276
6.3.3	VB-Skripte.....	1279
6.3.3.1	VB-Skripte exportieren	1279
6.3.3.2	VB-Skripte aus einem Ordner exportieren	1279
6.3.3.3	VB-Skripte importieren	1280
6.3.4	Textlisten	1282
6.3.4.1	Textlisten aus einem HMI-Gerät exportieren	1282
6.3.4.2	Textliste in ein HMI-Gerät importieren	1284
6.3.4.3	Erweiterte XML-Formate für Export/Import von Textlisten	1284
6.3.5	Grafiklisten	1287
6.3.5.1	Grafiklisten exportieren	1287
6.3.5.2	Grafikliste importieren.....	1287
6.3.6	Verbindungen.....	1288
6.3.6.1	Verbindungen exportieren	1288
6.3.6.2	Verbindungen importieren	1289
6.3.7	Bilder.....	1290
6.3.7.1	Übersicht der exportierbaren Bild-Objekte	1290
6.3.7.2	Alle Bilder eines HMI-Geräts exportieren	1294
6.3.7.3	Bild aus einem Bildordner exportieren	1297
6.3.7.4	Bilder in ein HMI-Gerät importieren	1299

6.3.7.5	Permanentfenster exportieren.....	1302
6.3.7.6	Permanentfenster importieren	1302
6.3.7.7	Alle Bildvorlagen eines HMI-Geräts exportieren.....	1303
6.3.7.8	Bildvorlagen aus einem Ordner exportieren	1307
6.3.7.9	Bildvorlagen importieren.....	1310
6.3.7.10	Exportieren eines Pop-up-Bilds	1314
6.3.7.11	Importieren eines Pop-up-Bildes	1316
6.3.7.12	Exportieren eines Slide-in-Bilds.....	1317
6.3.7.13	Importieren eines Slide-in-Bilds	1318
6.3.7.14	Bild mit Eingabemaske exportieren.....	1320
6.3.7.15	Bild mit Eingabemaske importieren	1322
6.4	Daten eines PLC-Geräts importieren/exportieren	1324
6.4.1	CFC-Pläne (Export/Import).....	1324
6.4.1.1	Export/Import von CFC-Plänen.....	1324
6.4.1.2	TIA Portal-Projektansicht: CFC-Pläne exportieren und importieren	1326
6.4.1.3	Export von CFC-Plänen	1328
6.4.1.4	Export nur von ausgewählten CFC-Plänen.....	1329
6.4.1.5	Import von CFC-Plänen	1331
6.4.1.6	Passwort für CFC-Pläne festlegen.....	1332
6.4.1.7	Passwort aus CFC-Plänen lesen.....	1333
6.4.1.8	Passwort für CFC-Pläne ändern	1335
6.4.1.9	Passwort aus CFC-Plänen löschen	1336
6.4.2	Bausteine	1338
6.4.2.1	XML-Struktur des Abschnitts Bausteinschnittstelle	1338
6.4.2.2	Änderungen des Objektmodells und Dateiformat XML	1348
6.4.2.3	DBs mit Schnappschüssen exportieren.....	1350
6.4.2.4	Bausteine mit Know-how-Schutz exportieren	1352
6.4.2.5	Export/Import von SCL-Bausteinen.....	1353
6.4.2.6	Export/Import strukturierter Typen von SCL-Bausteinen.....	1367
6.4.2.7	Export/Import von SCL-Aufrufbausteinen	1373
6.4.2.8	Mehrsprachige Kommentare in SCL exportieren/importieren	1390
6.4.2.9	F-Bausteine exportieren	1391
6.4.2.10	System-Bausteine exportieren	1391
6.4.2.11	GRAPH-Bausteine mit mehrsprachigem Text exportieren.....	1392
6.4.2.12	Baustein importieren	1393
6.4.2.13	Bausteine exportieren	1395
6.4.2.14	Bausteine/UDT mit offenem Verweis importieren	1402
6.4.2.15	Bausteine/UDT für Strukturänderungsobjekte importieren.....	1404
6.4.2.16	Export/Import des Unit-spezifischen Veröffentlichungsattributs von Bausteinen und Typen.	1405
6.4.2.17	Instanz-DB erstellen	1410
6.4.2.18	Auf DB-Wertparameter ohne Export zugreifen	1411
6.4.2.19	Export/Import von PLC-Meldetextlisten	1413
6.4.2.20	Dokumentinformation in eine SimaticML-Datei exportieren	1419
6.4.2.21	Export/Import von Meldungsinstanztexten	1420
6.4.2.22	Export/Import von Meldeklassen	1423
6.4.2.23	Export/Import von globalen Überwachungen des ProDiag-FB	1426
6.4.2.24	Export/Import von ProDiag-Überwachungen aus globalem Übersichtseditor.....	1428
6.4.2.25	Export/Import von Einstellungen für ProDiag-Überwachungen	1429
6.4.2.26	Beobachtungs- und Forcetabelle exportieren/importieren	1431
6.4.2.27	Anwenderdatentyp exportieren.....	1433
6.4.2.28	Anwenderdatentyp importieren	1434
6.4.2.29	Export von Daten im Format OPC UA XML	1435

6.4.2.30	Export/Import von UDT & DB	1436
6.4.2.31	Export/Import von Array- & Instanz-DB	1442
6.4.3	Technologieobjekte	1448
6.4.3.1	Übersicht über Technologieobjekte und Versionen	1448
6.4.3.2	XML-Struktur des Schnittstellenabschnitts Technologieobjekt	1450
6.4.3.3	Technologieobjekte exportieren	1453
6.4.3.4	Technologieobjekte importieren	1455
6.4.3.5	S7-1200 Motion Control	1456
6.4.3.6	S7-1500 Motion Control	1459
6.4.3.7	PID-Regelung	1471
6.4.3.8	Zählen	1473
6.4.3.9	Easy Motion Control	1473
6.4.4	Variablentabellen	1473
6.4.4.1	PLC-Variablentabellen exportieren	1473
6.4.4.2	PLC-Variablentabelle importieren	1474
6.4.4.3	Einzelne Variable oder Konstante aus einer PLC-Variablentabelle exportieren	1475
6.4.4.4	Einzelne Variable oder Konstante in eine PLC-Variablentabelle importieren	1477
6.5	Hardware-Daten importieren/exportieren	1478
6.5.1	AML-Dateiformat	1478
6.5.2	Pruned AML	1478
6.5.3	Übersicht über Objekte und Parameter von CAX-Import/-Export	1480
6.5.4	Struktur der CAX-Daten zum Import/Export	1482
6.5.5	AML-Typkennungen	1486
6.5.6	Exportieren/Importieren von Informationen zur Basiseinheit über AML	1489
6.5.7	Export/Import einer AML-Datei mit Verbindung zum Erweiterungsbaugruppenträger	1492
6.5.8	AML-Datei mit benutzerdefinierten GSD/GSDML-Attributen exportieren/importieren	1498
6.5.9	AML-Datei mit benutzerdefinierten Nicht-GSD/GSDML-Attributen exportieren/ importieren	1506
6.5.10	AML-Datei mit einsteckbarer Portkonfiguration exportieren/importieren	1527
6.5.11	AML-Datei mit IO-Link exportieren/importieren	1532
6.5.12	Verwalten von Verbindungen bei Erweiterungsbaugruppenträgern	1540
6.5.13	AML-Datei mit komplexer Variablenkonfiguration exportieren/importieren	1541
6.5.14	Export von CAX-Daten	1546
6.5.15	Import von CAX-Daten	1553
6.5.16	Export/Import von Submodulen	1557
6.5.17	AML-Dateien in einer UMAC-Umgebung exportieren/importieren	1570
6.5.18	AML-Datei mit normiertem Typbezeichner exportieren/importieren	1571
6.5.19	CAX-Daten ohne logische Adresse importieren	1571
6.5.20	Ausnahmen beim Import und Export von CAX-Daten	1579
6.5.21	Round-Trip-Geräte und -Module	1580
6.5.22	Export/Import-Topologie	1583
6.5.23	Import eines Geräts mit Bibliotheksreferenzen	1584
6.5.24	Export eines Geräteelements	1607
6.5.25	Import eines Geräteobjekts	1610
6.5.26	Export/Import eines Geräts mit festgelegter Adresse	1613
6.5.27	Export/Import eines Geräts mit Kanälen	1616
6.5.28	Export von Geräteelementobjekten	1618
6.5.29	Import von Geräteelementobjekten	1623
6.5.30	Export/Import von GSD/GSDML-basierten Geräten und Geräteelementen	1626
6.5.31	Exportieren/Importieren der Gerätekonfiguration mit virtueller Schnittstelle	1632
6.5.32	Exportieren/Importieren von Subnetzen	1635
6.5.33	Export/Import von IO-Systemen	1643

6.5.34	Exportieren/Importieren mehrsprachiger Kommentare	1645
6.5.35	Export/Import von ET200 SP/ET200 AL-Geräten mit Erweiterungsbaugruppenträgerverbindungen	1647
6.5.36	Exportieren/Importieren von PLC-Variablen.....	1659
6.5.37	Export/Import von RH/PLC	1662
6.5.38	Export/Import von AML mit benutzerdefinierten Variablen und Geräteelementen	1665
6.5.39	Import/Export fehlersicherer PLCs.....	1667
6.5.40	Import/Export fehlersicherer E/A.....	1674
6.5.41	Herstellerspezifisches Attribut exportieren/importieren	1676
6.5.42	AML-Attribute im Vergleich zu TIA Portal Openness-Attributen	1677
7	Wesentliche Änderungen	1681
7.1	Größere Änderungen für Langzeitstabilität in TIA Portal Openness V19	1681
7.2	Größere Änderungen für Langzeitstabilität in TIA Portal Openness V17	1683
7.3	Größere Änderungen in TIA Portal Openness V16	1688
7.4	Größere Änderungen in TIA Portal Openness V15.1	1689
7.5	Größere Änderungen in TIA Portal Openness V15	1692
7.6	Die wichtigsten Änderungen in V14 SP1	1695
7.6.1	Die wichtigsten Änderungen in V14 SP1	1695
7.6.2	Wesentliche Änderungen im Objektmodell	1698
7.6.3	Änderungen an der Pilotfunktionalität.....	1702
7.6.4	Änderungen bei Export und Import	1707
7.6.4.1	Änderungen bei Export und Import	1707
7.6.4.2	Änderungen in der API	1707
7.6.4.3	Schemaerweiterung.....	1709
7.6.4.4	Schemaänderungen.....	1711
7.6.4.5	Verhaltensänderungen.....	1714
7.6.4.6	Änderungen an Bausteinattributen.....	1725
7.7	Die wichtigsten Änderungen in V14	1727
7.7.1	Wesentliche Änderungen des Objektmodells	1727
7.7.2	Vor dem Hochrüsten einer Anwendung auf TIA Portal Openness V14.....	1729
7.7.3	Wesentliche Stringänderungen.....	1730
7.7.4	Importieren von Dateien, die mit TIA Portal Openness V13 SP1 und älter erzeugt wurden.	1733
	Index.....	1737

Hinweise zur Cybersecurity

Hinweise zur Cybersecurity

Siemens bietet Produkte und Lösungen mit Industrial Cybersecurity-Funktionen an, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen.

Um Anlagen, Systeme, Maschinen und Netzwerke vor Cyberbedrohungen zu schützen, ist es notwendig, ein ganzheitliches, modernes Industrial Cybersecurity-Konzept umzusetzen und kontinuierlich zu pflegen. Die Produkte und Lösungen von Siemens bilden nur ein Element eines solchen Konzepts.

Der Kunde ist dafür verantwortlich, unbefugten Zugang zu seinen Anlagen, Systemen, Maschinen und Netzwerken zu verhindern. Solche Systeme, Maschinen und Komponenten dürfen nur im Bedarfsfall, im erforderlichen Umfang und bei entsprechenden Sicherheitsmaßnahmen (z. B. Firewalls und/oder Netzwerksegmentierung) mit einem Unternehmensnetzwerk oder dem Internet verbunden werden.

Weiterführende Informationen zu möglichen Schutzmaßnahmen im Bereich Industrial Cybersecurity finden Sie unter .

<https://www.siemens.com/global/en/products/automation/topic-areas/industrial-cybersecurity.html> (<https://www.siemens.com/global/en/products/automation/topic-areas/industrial-cybersecurity.html>)

Die Produkte und Lösungen von Siemens werden ständig weiterentwickelt, um sie sicherer zu machen. Siemens empfiehlt, Produkt-Updates bei Verfügbarkeit unbedingt anzuwenden und immer die neuesten Produktversionen zu benutzen. Die Verwendung von nicht mehr unterstützten Produktversionen und die Nicht-Anwendung der neuesten Updates kann das Risiko des Kunden gegenüber Cyberbedrohungen erhöhen.

Um stets über Produkt-Updates informiert zu sein, melden Sie sich für den Siemens Industrial Cybersecurity RSS-Feed an unter

<https://new.siemens.com/de/de/produkte/services/cert.html> (<https://new.siemens.com/global/en/products/services/cert.html>)

Liesmich zu TIA Portal Openness

2.1 Abgekündigte Features in TIA Portal Openness

Einleitung

Im Allgemeinen verfolgt TIA Portal Openness eine Strategie von langfristig stabilen APIs. Es gibt allerdings vereinzelte Situationen, in denen APIs überarbeitet werden müssen:

Terminologie:

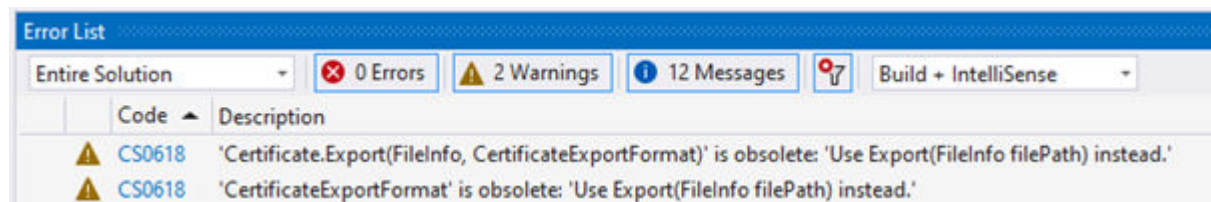
Produktrelease	Version TIA Portal Openness	
TIA Portal V18	API V15.1	Alte API
	API V16	Alte API
	API V17	Alte API
	API V18	Neue API
TIA Portal V19	API V16	Alte API
	API V17	Alte API
	API V18	Alte API
	API V19	Neue API

Statische Teile von Objektmodellen

Abgekündigte Features werden für eine bestimmte API-Version bekannt gegeben und in der darauffolgenden Produktversion entfernt. Die alten APIs im nächsten Produktrelease sind davon nicht betroffen.

Bei statischen Teilen von Objektmodellen werden abgekündigte Features als **obsolete** gekennzeichnet. Bei Verwendung dieser Features im Code wird eine Compiler-Warnung ausgelöst, sie funktionieren aber weiterhin in der betreffenden Version.

Beispiel für eine Compiler-Warnung:



2.1 Abgekündigte Features in TIA Portal Openness

Beispiel für hervorgehobene obsoleete Features im Code:

```
private static void ExportCertificate(Siemens.Engineering.Security.Certificate certificate)
{
    FileInfo fileInfo = new FileInfo($"C:\\Temp\\{certificate.SubjectCommonName}.cer");
    certificate.Export(fileInfo, Siemens.Engineering.Security.CertificateExportFormat.Cer);
}
```

Es wird empfohlen, abgekündigten Code wie folgt anzupassen:

Abgekündigt in API V18

- Klasse: Siemens.Engineering.AddIn.VersionControl.InitialPreExportInfo
Abgekündigt und ersetzt durch Siemens.Engineering.AddIn.VersionControl.PreExportInfo
- Klasse: Siemens.Engineering.AddIn.VersionControl.InitialPostExportInfo
Abgekündigt und ersetzt durch Siemens.Engineering.AddIn.VersionControl.PostExportInfo
- Klasse: Siemens.Engineering.AddIn.VersionControl.SyncPreExportInfo
Abgekündigt und ersetzt durch Siemens.Engineering.AddIn.VersionControl.PreExportInfo
- Klasse: Siemens.Engineering.AddIn.VersionControl.SyncPostExportInfo
Abgekündigt und ersetzt durch Siemens.Engineering.AddIn.VersionControl.PostExportInfo
- Klasse: Siemens.Engineering.AddIn.VersionControl.VciInitialExportAddInContext
Abgekündigt und ersetzt durch Siemens.Engineering.AddIn.VersionControl.ExportContext
- Klasse: Siemens.Engineering.AddIn.VersionControl.VciSyncExportAddInContext
Abgekündigt und ersetzt durch Siemens.Engineering.AddIn.VersionControl.ExportContext
- Klasse: Siemens.Engineering.AddIn.VersionControl.VciRepositoryAddIn
Abgekündigt und ersetzt durch
Siemens.Engineering.AddIn.VersionControl.VciWorkspaceRepositoryAddIn
- Klasse: Siemens.Engineering.AddIn.VersionControl.VciRepositoryAddInProvider
Abgekündigt und ersetzt durch
Siemens.Engineering.AddIn.VersionControl.VciWorkspaceRepositoryAddInProvider
- Klasse: Siemens.Engineering.AddIn.VersionControl.VciInitialExportSupport
Abgekündigt und ersetzt durch
Siemens.Engineering.AddIn.VersionControl.ExportWorkflowSupport
- Klasse: Siemens.Engineering.AddIn.VersionControl.VciSyncExportSupport
Abgekündigt und ersetzt durch
Siemens.Engineering.AddIn.VersionControl.ExportWorkflowSupport
- Klasse: Siemens.Engineering.AddIn.VersionControl.VciWorkflowAddInSupport
Abgekündigt und ersetzt durch
Siemens.Engineering.AddIn.VersionControl.VciWorkspaceRepositoryWorkflowAddIn
- Klasse: Siemens.Engineering.HmiUnified.RuntimeSettings.HmiRuntimeSetting

Abgekündigte Eigenschaft	Umgezogen
OperateAsOpcServer { get; set; }	In Klasse Siemens.Engineering.HmiUnified.RuntimeSettings.HmiOpcUaServerRuntimeSettings verschoben

- Klasse: Siemens.Engineering.HmiUnified.UI.Parts.HmiTrendPartBase

Abgekündigte Eigenschaft	Umgezogen
TrendMode { get; set; }	Von Basisklasse zu abgeleiteten Klassen Siemens.Engineering.HmiUnified.UI.Parts.HmiTrendPart and Siemens.Engineering.HmiUnified.UI.Parts.HmiFunctionTrendPart verschoben

- Klasse: Siemens.Engineering.SIVArc.ScreenRule

Abgekündigte Eigenschaft	Abgekündigt
ScreenLibraryItem { get; set; }	Ersatzlos abgekündigt.

Abgekündigt in API V19

- Klasse: Siemens.Engineering.HW.MulticastableTransferArea

Abgekündigte Eigenschaft	Ersetzt durch
Siemens.Engineering.HW.Address Address { get; }	Siemens.Engineering.HW.AddressComposition Addresses { get; }

- Klasse: Siemens.Engineering.Security.Certificate

Abgekündigte Methode	Ersetzt durch
System.Void Export(System.IO.FileInfo filePath, Siemens.Engineering.Security.CertificateExportFormat exportFormat)	System.Void Export(System.IO.FileInfo filePath)

- Enum: Siemens.Engineering.Security.CertificateExportFormat
Ersatzlos abgekündigt
- Enum: Siemens.Engineering.HW.WebserverUserPermissions2
Ersatzlos abgekündigt
- Enum: Siemens.Engineering.HW.MaximumBufferedReceivedFrames
Ersatzlos abgekündigt
- Enum: Siemens.Engineering.HW.PortType
Abgekündigt und ersetzt durch Siemens.Engineering.HW.PortConfiguration
- Ausnahmeklasse: Siemens.Engineering.SimaticMLVersionNotSupportedException
Ersatzlos abgekündigt

Dynamische API-Teile

Abgekündigte Features werden für eine bestimmte TIA Portal-Version bekannt gegeben und in der darauffolgenden TIA Portal-Version entfernt. Betroffen sind die alten APIs und die neue API des nächsten Produktrelease.

Bei dynamischen API-Teilen kann aus technischen Gründen keine Compiler-Warnung erzeugt werden. Beachten Sie zu abgekündigten Features in diesem Bereich das Systemhandbuch.

Beispiel für einen dynamischen API-Teil:

```
private static ulong GetPortValue(Siemens.Engineering.HW.DeviceItem item)
{
    const string attributeName = "PortType";
    ulong attributeValue = (ulong)item.GetAttribute(attributeName);
    return attributeValue;
}
```

Es wird empfohlen, abgekündigten Code wie folgt anzupassen:

Abgekündigt in TIA Portal V18 (alle API-Versionen)

Hardware-Parameter in 6ES7148-6JG00-0BB0/V5.1

Abgekündigter Attributname	Ersetzt durch
PortType	PortConfiguration

2.2 Liesmich

Sicherheitsmaßnahmen für TIA Portal Openness-Anwendungen

Es ist empfehlenswert,

- eine TIA Portal Openness-Anwendung mit Administratorrechten im Ordner "Programme" zu installieren.
- das dynamische Laden von Programmbestandteilen wie Assemblies oder DLLs aus dem Benutzerbereich zu vermeiden.
- die TIA Portal Openness-Anwendung mit Benutzerrechten auszuführen.

Hardware-Parameter

Eine Beschreibung der Hardware-Parameter steht im Installationsordner von TIA Portal unter Siemens\Automation\Portal V*\PublicAPI\V*\HW Parameter Description\Openness_hardware_parameter_description.pdf zur Verfügung.

Attribute können Abhängigkeiten haben und einander beeinflussen. Wenn ein Attribut eines Geräteelements oder Kanals geändert wurden, müssen deshalb alle Attribute des Geräteelements oder Kanals überprüft werden.

Hinweis

V* bezieht sich auf den entsprechend der installierten Version von TIA Portal angepassten Pfad.

Kopieren einer TIA Portal Openness-Anwendung

Wenn Sie eine ausführbare TIA Portal Openness-Anwendung kopieren, kann es unter bestimmten Umständen vorkommen, dass der Verzeichnispfad, in dem die TIA Portal Openness-Anwendung ursprünglich erzeugt wurde, von der TIA Portal Openness-Anwendung ausgelesen wird.

Abhilfe:

Wenn Sie die TIA Portal Openness-Anwendung in ein neues Verzeichnis kopiert haben, öffnen und schließen Sie den Dialog "Eigenschaften", um den Windows-Cache zu aktualisieren.

Unterstützung bestimmter Funktionen in einem TIA Portal-Projekt

Fehlersicherheit

Wenn Sie mit TIA Portal Openness arbeiten, sind gewisse Einschränkungen in Bezug auf die Fehlersicherheit zu beachten. Weitere Informationen finden Sie in der Dokumentation "SIMATIC Sicherheitssysteme - Konfiguration und Programmierung".

Test Suite

Die Unterstützung von TIA Portal Openness für die Test Suite ist in der jeweiligen Online-Hilfe zu Test Suite V19 beschrieben.

Verbesserung der Leistung von TIA Portal Openness

Um die maximale Leistung von TIA Portal Openness zu erreichen, können Sie die globale Suchfunktion des TIA Portals ausschalten. Zum Ausschalten der globalen Suche nutzen Sie die Benutzeroberfläche oder den TIA Portal Openness API-Aufruf. Wenn die TIA Portal Openness-Anwendung fertig ist, kann die globale Suche wieder eingeschaltet werden. Obwohl sich die Leistung durch das Ausschalten der globalen Suche verbessert, arbeiten alle TIA Portal Openness-Funktionen auch mit eingeschalteter globaler Suche einwandfrei.

Threadsicherer Programmcode

Achten Sie darauf, dass Ihr Code threadsicher ist: Ein Ereignis erscheint in verschiedenen Threads.

Exportverhalten von Bildelementen mit aktiviertem Style

Beim Export eines Bildelements mit aktiviertem Style werden nicht die Attribute des Style-Elements exportiert, sondern nur die Attribute des Bildelements vor Aktivierung des Styles. Wenn ein Style ausgewählt wird und "UseDesignColorSchema" für das Bildelement aktiviert ist, übernimmt das Bildelement auf der Benutzeroberfläche die Attributwerte des Styles. In der Datenbank werden für dieses Bildelement jedoch noch immer die Attributwerte gespeichert, die vor Auswahl des Styles festgelegt waren. TIA Portal Openness exportiert diese tatsächlichen, in der Datenbank gespeicherten Werte.

Wird der Style deaktiviert und wieder aktiviert und das Bildelement dann erneut exportiert, werden für dieses Bildelement die im Style-Element geltenden Attributwerte exportiert. Die Attributwerte des für dieses Bildelement ausgewählten Style-Elements werden in der Datenbank gespeichert, wenn "UseDesignColorSchema" nicht aktiviert ist.

Dieses Problem kann wie folgt behoben werden:

1. Weisen Sie das Bildelement dem Style-Element zu:
 - Die Datenbank enthält die Attributwerte, die vor Aktivierung des Styles galten.
 - Die Benutzeroberfläche übernimmt Attribute direkt vom Style-Element.
2. Exportieren Sie das dem Style-Element zugewiesene Bildelement:
 - Die XML-Datei enthält die Attributwerte aus der Datenbank, die den Werten vor Aktivierung des Styles entsprechen.
3. Deaktivieren Sie "UseDesignColorSchema":
 - Die Attributwerte des Style-Elements werden in der Datenbank den Attributen des Bildelements hinzugefügt.
4. Aktivieren Sie "UseDesignColorSchema":
 - Die Attributwerte des Bildelements in der Datenbank werden nicht verändert und entsprechen noch denen aus Schritt 3.
 - Die Benutzeroberfläche übernimmt Attribute direkt vom Style-Element.
5. Exportieren Sie das dem Style-Element zugewiesene Bildelement:
 - Die XML-Datei enthält die Attributwerte aus der Datenbank, die in Schritt 3 festgelegt wurden und den Werten des Style-Elements entsprechen.

Importieren von ASi-Slaves über AML

Wird einer der folgenden ASi-Slaves über eine AML-Datei importiert, wird die Firmware-Version des Geräteelements in jedem Fall auf V13.0 gesetzt:

- ASIsafe FS400 RCV-B: 3SF7 844-*B***-***1
- ASIsafe FS400 RCV-M: 3SF7 844-*M***-***1
- ASIsafe FS400 TRX-M: 3SF7 844-*M***-***T0
- ASIsafe FS400 RCV-C: 3SF7 844-*T***-***1

Exportieren und Importieren von Funktionstasten

Funktionstasten werden während des Imports synchronisiert. Wenn eine Funktionstaste im globalen Bildschirm erzeugt wird und die Taste im Bildschirm leer ist, nutzt die entsprechende Funktionstaste die globale Definition in allen Bildschirmen.

Wenn Sie die globale Nutzung von Funktionstasten nach dem Import deaktivieren möchten, definieren Sie leere Tasten in den Bildschirmen und importieren die Bildschirmtypen in der folgenden Reihenfolge: Globaler Bildschirm, Vorlagen, Bildschirme.

Wenn Sie beim Exportieren von Bildschirmen sicherstellen möchten, dass die globale Definition einer Funktionstaste nicht von der Vorlage oder vom globalen Bildschirm genutzt wird, erzeugen Sie eine leere Funktionstaste in dem Bildschirm. Wählen Sie die erforderliche Funktionstaste im Bildschirm. Aktivieren Sie anschließend die Eigenschaft "Use global assignment" und deaktivieren Sie diese wieder.

Zugriff auf ein Gerät, das online ist

Das Schreiben von Attributen in ein Gerät, das online ist, wird nicht unterstützt. Das Lesen von Attributen wird unterstützt.

Das Trennen von einem Subnetz, während das Gerät online ist, wird nicht unterstützt.

Instanzspezifische Attribute beim Importieren von Bausteinen über TIA Portal Openness

In bestimmten Situationen können die Importregeln den Verlust von instanzenspezifischen Attributen wie zum Beispiel Anlaufwerten bedeuten.

Prioritätsattribut für Schreibzugriff für den OB-Baustein

Der Name des Prioritätsattributs für Schreibzugriff für den OB-Baustein wurde in PriorityNumber geändert.

Informationen zu bestimmten Funktionen

Beachten Sie die FAQ-Einträge beim Siemens Industry Online Support für weitere Informationen zu den folgenden Openness-Funktionen:

- Projekt archivieren/wiederherstellen
- Beobachtungstabellen exportieren/importieren

Siehe auch

Test Suite V18 (<https://support.industry.siemens.com/cs/ww/de/view/109813414>)

2.3 Größere Änderungen für Langzeitstabilität in TIA Portal Openness V19

Änderungen

Wenn Sie die Hinweise zur versionsübergreifenden Programmierung beachtet haben und Ihre Openness-Anwendung nicht auf V19 aufbauen, läuft Ihre Anwendung ohne jede Einschränkung auf jedem Rechner, auch wenn nur ein TIA Portal V19 installiert ist.

Wenn Sie Ihre Openness-Anwendung auf V19 aufbauen, ist es notwendig, Ihre Anwendung mit der Siemens.Engineering.dll von V19 neu zu übersetzen. In manchen Fällen kann es erforderlich sein, den Code Ihrer Anwendung anzupassen.

CAX/AML-Datenaustausch

Änderungen im Verhalten der Befehlszeilenfunktion

Ab TIA Portal V19 ist die Befehlszeilenfunktion für den CAX-Export und -Import für den Benutzer nicht mehr verfügbar.

Änderung im Verhalten des PROFINET/Ethernet-Port-Handlings

Ab TIA Portal V19 wird der Labelwert für PROFINET/Ethernet-Schnittstellenports ohne Leerzeichen exportiert, unabhängig von seinem Wert in TIA Portal. Beispiel: Ist das Label = P1 R, hätte die AML-Datei das Label = P1R

Änderung im Verhalten des IO Link-Port-Handlings

Ab TIA Portal V18 Update 2 wird der IO-Link-Port mit einem geänderten Labelwert 'C/Q<n>' ausgetauscht, wobei n die Portnummer ist. Beispiel: C/Q1, C/Q2 usw.
Verwenden Sie für den AML-Austausch der IO-Link-Konfiguration in TIA Portal V18 Update 2 (über S7-PCT) die Version "S7-PCT 3.5 SP3 Update 3" oder höher.

WebServerUserManagement

Änderungen im Verhalten beim Entfernen/Deaktivieren von vorhandenen Openness-Diensten

Bei TIA Portal V19 ist die zentralisierte UMAC-Funktion aktiv, aber WebServerUserManagement ist für alle PLCs mit einer Firmware-Version ab V3.1 deaktiviert.

Bis TIA Portal V18 ist jedoch WebServerUserManagement betriebsbereit, während die zentralisierte UMAC-Funktion für PLCs mit einer Firmware-Version ab V3.0 inaktiv ist.

UMAC auf PLC

Änderungen im Verhalten eines Online-Legitimationsaufrufs über UMAC auf PLC

Wenn die Openness-Anwendung mit einer TIA Portal Openness API <= 18 läuft, ist die Online-Legitimation über Openness über die Download- und Upload-Konfiguration noch möglich. Dies gilt nur, wenn die verwendete PLC durch eine alte ProtectionLevel-Legitimation geschützt ist.

Bei TIA Portal Openness API $\geq$ V19 werden alle Legitimationsaufrufe, insbesondere das neue UMAC auf PLC, dem Event Handler des Online-Legitimationseignisses der ConnectionConfiguration-Klasse bereitgestellt. Sollte der Typ OnlineAuthenticationConfiguration dort nicht vom Benutzercode behandelt werden, erfolgt ein zweiter Aufruf der Rückrufmethode der jeweiligen Funktion (z. B. Download oder Upload). Dort können jedoch nur die alten Schutzmechanismen (bis V18) verarbeitet werden.

Durchsetzung strenger Passwort-Richtlinien

Änderungen im Verhalten bei strengen Passwort-Richtlinien für den Know-how-Schutz von Bausteinen

Bei TIA Portal V19 werden strenge Passwort-Richtlinien bei der Einstellung eines Passworts angewandt. Dies gilt auch für Aufrufe von TIA Portal Openness, insbesondere für die Einstellung des Know-how-Schutzes. Andernfalls wird eine Ausnahme ausgelöst, wenn das Passwort die Richtlinie nicht erfüllt.

WinCC Unified-Bildeditor

Änderungen im Verhalten von MultilingualText-Elementen im Rahmen des WinCC Unified-Bildeditors

Alle MultilingualText-Elemente werden im Rahmen des WinCC Unified-Bildeditors seit TIA Portal V18 von unformatiert in formatiert geändert. Das bedeutet, dass alle Texte von nun an formatiert gesetzt werden müssen. Klartext wird durch Auslösen einer Ausnahme abgelehnt. Beispiel:

```
Language language = project.LanguageSettings.Languages.Find(new CultureInfo("en-US"));
MultilingualText multilingualToolTipText1 = ((HmiButton)screenItem1).ToolTipText;
MultilingualTextItem multilingualTextItem1 = multilingualToolTipText1.Items.Find(language);
multilingualTextItem1.Text = "<body><p>Modified button text from Openness</p></body>";
```

Unterstützung von Bausteinen/UDTs

Änderungen im Schema, um die Verwendung von NamedValueTypes in Bausteinen zu unterstützen

Im TIA Portal V19 wurde die Unterstützung für genannten Werttypen eingeführt (nur für Software Units in S7-1500 PLCs). In Openness wird ein neuer Gültigkeitsbereich "NamedValueConstant" im SimaticML-Schema ab V19 eingeführt, um die Verwendung eines genannten Werttyps in Programmbausteinen oder PLC-Datentypen zu unterstützen. Die genannten Werttypen, die in PLC-Programmierartefakten (Programmbausteinen, PLC-

2.4 Hinweise zum Schreiben von langfristig stabilem Code

Datentypen) verwendet werden, sind im neuen Gültigkeitsbereich "NamedValueConstant" beim Importieren/Exportieren über SimaticML sichtbar.

```
<Access Scope="NamedValueConstant" UId="27">  
<Constant Name="_.siemens.simatic.Named_value_type_1#UNDEFs" UId="28"/>  
</Access>  
<Token Text=";" UId="31"/>  
<NewLine Num="2" UId="32"/>
```

2.4 Hinweise zum Schreiben von langfristig stabilem Code

Versionswechsel

Wenn Sie einige Hinweise zum Schreiben von langfristig stabilem Code beachten, werden Sie Ihre Anwendung mit anderen Versionen des TIA Portals nutzen können, ohne den Code Ihrer Anwendung zu ändern.

Hinweis

V* und *.ap* im Dokument beziehen sich auf den entsprechend der installierten Version des TIA Portals angepassten Pfad und die Erweiterung.

Registry-Pfad und Datei appconfig

Modifikationen sind notwendig, um den Registry-Pfad und die Konfigurationsdatei der Anwendung zu ändern. Beispiel:

"C:\Programme\Siemens\Automation\Portal V14\PublicAPI\V14 SP1\Siemens.Engineering.dll"
muss geändert werden in

"C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Siemens.Engineering.dll".

Um langfristig stabilen Code zu schreiben, muss der Registry-Pfad konfigurierbar sein und die Konfigurationsdatei der Anwendung muss aktualisiert werden.

Installationspfad

Modifikationen sind notwendig, um den Installationspfad des TIA Portals zu ändern. Beispiel:
"C:\Programme\Siemens\Automation\Portal V14\PublicAPI\V14 SP1\Siemens.Engineering.dll"
muss geändert werden in

"C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Siemens.Engineering.dll".

Um langfristig stabilen Code zu schreiben, muss der Installationspfad konfigurierbar sein.

Erweiterungen von Projektdateien und Bibliotheken des TIA Portals

Modifikationen sind notwendig, um die Erweiterungen von Projektdatei und Bibliotheken des TIA Portals zu ändern. Beispiel:

*.ap14

muss geändert werden in

.ap.

Um langfristig stabilen Code zu schreiben, müssen Erweiterungen von Projektdateien und Bibliotheken des TIA Portals konfigurierbar sein.

Öffnen eines Projekts

Um langfristig stabilen Code zu schreiben, muss die Methode `Projects.OpenWithUpgrade` statt der Methode `Projects.Open` verwendet werden.

Hierarchie der Ergebnisse

Die Hierarchie und/oder die Reihenfolge von Ergebnisklassen, z. B. beim Übersetzen von Ergebnissen, kann sich versionsübergreifend ändern.

Um langfristig stabilen Code zu schreiben, müssen Sie es vermeiden, Vermutungen über die Tiefe und Reihenfolge bestimmter Ergebnisse anzustellen.

Das Kategorienlayout ist darauf ausgelegt, hinsichtlich der eindeutigen Typnamen `CompilerResult`, `CompareResult`, `DownloadResult`, `UploadResult` langfristig stabil zu sein. Es gibt auch eine neue Ergebnisklasse: `TransferResult`. Inhalt, Hierarchie und Reihenfolge folgen der Anzeige auf der TIA Portal-Benutzeroberfläche des aktuell ausgeführten oder installierten TIA Portals.

Was ist neu in TIA Portal Openness?

Kompatibilität und Langzeitstabilität

- **Siemens.Engineering.dll-Assemblies**
Da die Siemens.Engineering.dll-Assemblies der Versionen V16, V17 und V18 im Lieferumfang enthalten sind, laufen Anwendungen, die auf V16, V17 und V18 basieren, auch in V19 ohne Änderung. Um die Funktionen von V19 nutzen zu können, müssen Sie die DLL von V19 integrieren und die Anwendung neu übersetzen.
Die "Siemens.Engineering.dll"-Assemblies befinden sich im Installationsverzeichnis unter "PublicAPI\[version]\". Beispielsweise ist die dll-Datei für V19 unter "C:\Program Files\Siemens\Automation\Portal V*\PublicAPI\V19\Siemens.Engineering.dll" zu finden.

Hinweis

Die V15.1 Siemens.Engineering.dll ist nicht als Teil von TIA Portal Openness V19 erhältlich.

- **Exportieren von SIMATIC ML-Dateien**
Die "Siemens.Engineering.dll"-Assemblies der Versionen V16, V17 und V18 erzeugen SIMATIC ML-Dateien der Version V19 von TIA Portal.
- **Importieren von SIMATIC ML-Dateien**
Jede Version der "Siemens.Engineering.dll"-Assemblies kann SIMATIC-ML-Dateien aus der gleichen Version und aus jeder unterstützten Version des vorherigen Release importieren.
Beispiel: SIMATIC ML-Dateien der Version V18 können mit den "Siemens.Engineering.dll"-Assemblies der Version V19 importiert werden.

Hinweis

Die Versionsnummer V* bezieht sich auf die installierte Version der TIA Portal Openness API.

Weitere Informationen zu Änderungen am Objektmodell finden Sie unter TIA Portal Openness-Objektmodell (Seite 55).

Neue Funktionen

Die folgenden neuen Funktionen und Innovationen sind in TIA Portal Openness V19 verfügbar. Weitere Einzelheiten zu den verschiedenen Themen finden Sie in den einzelnen Abschnitten der Produktdokumentation.

- **Allgemein**
 - Importieren jeder unterstützten SimaticML-Version, unabhängig davon, welche API-Version verwendet wird.
 - Flexiblerer SimaticML-Import bei Optionen mit Projektsprachen.
- **Projekte**
 - Konfiguration der Verwendung von S7-1500-Bausteinen in virtuellen PLCs

- Hardwarekonfiguration
 - Erstellen, Lesen, Aktualisieren und Löschen von Transferbereichen für den direkten Datenaustausch zwischen PLCs.
 - Verwaltung der Konfiguration der PLC-Systemprotokolle.
 - Verwaltung der Konfiguration der neuen PLC-Zugriffskontrolle für CPU-Firmware 3.1 (UMAC und Zugriffsstufe).
 - Änderung der voreingestellten Sprache für OPC UA-Alarme und -Ereignisse bei der mehrsprachigen PLC-Unterstützung.
 - Massenänderung von Hardware-Parametern in der richtigen Reihenfolge in allen überlasteten "SetAttributes"-Methoden
- Hardware-Bausteine
 - Neu unterstützte Module für den Parameterzugriff: ET200pro Safety, ET200eco PN Safety, ET200MP Safety.
 - Erweiterter Parameterzugriff für SCALANCE XC-200 (ab V4.3), XP-200 (ab V4.3), SC-600 (ab V2.3).
- CAX-Datenaustausch
 - Importieren/Exportieren von Cax-Daten über AutomationML und Abrufen von Ergebnissen über API anstatt einer Protokolldatei.
 - Unterstützung zusätzlicher Attribute zum Austausch von Hardware-Konfigurationen.
- Online-Szenarien
 - Erhalt von zugänglichen Online-Geräten zum Download oder Upload.
 - Herunterladen von PLC, einschließlich Sicherheit in den SIMATIC-Speicherkartenordner.
 - Verarbeitung von UMAC-Benutzerverwaltungsdaten zum Download.
 - Bereitstellen von UMAC-Anmeldedaten zur PLC-Zugriffslegitimation online.
- PLC-Anwenderprogramme
 - Lese- und Schreibzugriff für zusätzliche Spalten in Datenbausteinen.
 - Informatives Attribut auf Programmbausteinen zur Unterstützung für virtuelle PLCs.
 - Erweiterung des SimaticML-Schemas zur Unterstützung der Verwendung von genannten Werttypen.
 - Genannte Wertkonstanten werden beim Export/Import von Bausteinen unterstützt.
- Safety-Einstellungen
 - Konfiguration von Seriennummern zur eindeutigen Identifizierung einer F-PLC im Safety Administration Editor.
- UMAC
 - Konfiguration von Alias-Namen für Projektnutzer.
- Technologieobjekte
 - Unterstützung von Gruppen für Technologieobjekte.
 - Import und Export von Dateien für Interpretierprogramme

- Startdrive
 - Verbindung eines Technologieobjekts mit einem Startdrive-Telegramm.
 - Lesen einer Hardware-ID eines Startdrive-Telegramms.
 - Unterstützung neuer Startdrive-Geräte.
 - Parametrierung von Gebern aus Fremdfertigung.
- WinCC Unified
 - Zugriff auf zusätzliche Objektattribute und Runtime-Einstellungen.
- Test Suite Advanced
 - Konfiguration von OPC UA-Einstellungen für Systemtests.
 - Konfiguration der Betriebsart für Anwendungsfalltests.
 - Unterstützung für Masterkopien.
- Version Control Interface (VC)
 - VCI-Einstellung zur Verwaltung nicht aktiver Projektsprachen während des SimaticML-Imports.
 - Initialisieren des Status des VCI-Objekts auf die abgebildete Datei.
- Meldungen und ProDiag
 - Verwaltung der Systemdiagnose bei der PLC-Konfiguration.
- Webserver
 - Verwaltung des Datenzugriffs für den Webserver für CPU-Firmware 3.1.

Siehe auch

Größere Änderungen in TIA Portal Openness V16 (Seite 1688)

Größere Änderungen für Langzeitstabilität in TIA Portal Openness V17 (Seite 1683)

Größere Änderungen für Langzeitstabilität in TIA Portal Openness V19 (Seite 30)

Größere Änderungen in TIA Portal Openness V15.1 (Seite 1689)

Größere Änderungen in TIA Portal Openness V15 (Seite 1692)

Die wichtigsten Änderungen in V14 SP1 (Seite 1695)

Die wichtigsten Änderungen in V14 (Seite 1727)

Grundlagen

4.1 Installation

4.1.1 Voraussetzungen für TIA Portal Openness

Voraussetzungen für den Einsatz von TIA Openness-Anwendungen

- Auf dem PC ist ein auf dem TIA Portal basierendes Produkt installiert, zum Beispiel "STEP 7 Professional" oder "WinCC Professional".
- "TIA Portal Openness" ist auf dem PC installiert.
Siehe TIA Portal Openness installieren (Seite 40)

Unterstützte Windows-Betriebssysteme

Die folgende Tabelle zeigt, welche Kombinationen aus Betriebssystem Windows, TIA Portal und Benutzeranwendung miteinander kompatibel sind:

Betriebssystem Windows	TIA Portal	Benutzeranwendung
64-Bit	64-Bit	32-Bit, 64-Bit und beliebige CPU

Voraussetzungen für die Programmierung von TIA Portal Openness-Anwendungen

- Microsoft Visual Studio 2017 oder höher mit .Net Framework SDK 4.8 und dem Windows Classic Desktop package

Notwendige Kenntnisse des Benutzers

- Kenntnisse als Systemtechniker
- Fortgeschrittene Kenntnisse in Microsoft Visual Studio 2017 oder höher mit .Net Framework SDK 4.8
- Fortgeschrittene Kenntnisse in C# / VB.net und .Net Framework
- Benutzerkenntnisse des TIA Portals

TIA Portal Openness-Remoting-Kanäle

Die TIA Portal Openness-Remoting-Kanäle sind als Typ `IpcChannel` registriert, wobei der Parameter `"ensureSecurity"` auf `"false"` gesetzt ist.

Hinweis

Vermeiden Sie, einen weiteren `IpcChannel` mit dem Parameterwert `"ensureSecurity"` ungleich `"false"` und einer Priorität höher als oder gleich `"1"` zu registrieren.

Für den `IpcChannel` sind die folgenden Attribute festgelegt:

Attribut	Einstellungen
"name" und "portName"	Auf <code>"\${Process.Name}_{Process.Id}"</code> oder <code>"\${Process.Name}_{Process.Id}_{AppDomain.Id}"</code> gesetzt, wenn in einer anderen als der Standard-AppDomain der Anwendung registriert.
"priority"	Auf den Standardwert <code>"1"</code> gesetzt.
"typeFilterLevel"	Auf <code>"Voll"</code> gesetzt.
"authorizedGroup"	Auf die Zeichenkette des NT-Kontowerts für das integrierte Benutzerkonto (d.h. jeder) gesetzt.

Siehe auch

Benutzer der Benutzergruppe "Siemens TIA Openness" hinzufügen (Seite 41)

4.1.2 TIA Portal Openness installieren

Einführung

Installiert wird "TIA Portal Openness" über das Kontrollkästchen für TIA Portal Openness (unter Optionen) bei der Installation des TIA Portals.

Voraussetzungen

- Hardware und Software des Programmiergeräts oder PCs erfüllen die Systemvoraussetzungen.
- Sie haben Administratorrechte.
- Laufende Programme wurden geschlossen.
- Die Autorun-Funktion ist deaktiviert.
- WinCC und/oder STEP 7 ist/sind installiert.
- Die Versionsnummer von "TIA Portal Openness" entspricht der Versionsnummer von WinCC und STEP 7.

Hinweis

Wenn bereits eine Vorgängerversion von TIA Portal Openness installiert ist, wird die aktuelle Version parallel installiert.

Vorgehensweise

Zum Installieren von TIA Portal Openness stellen Sie sicher, dass das Kontrollkästchen für TIA Portal Openness während der Installation des TIA Portals ausgewählt ist. Befolgen Sie die nachstehenden Schritte, um die TIA Openness-Installation zu prüfen.

1. Wählen Sie im Menü "Konfiguration" den Ordner "Optionen" aus.
2. Aktivieren Sie das Kontrollkästchen für TIA Portal Openness.
3. Klicken Sie auf "Next" und wählen Sie die erforderliche Option.

Befolgen Sie den Installationsvorgang von TIA Portal, um die Installation von TIA Portal Openness abzuschließen.

Ergebnis

TIA Portal Openness ist auf dem PC installiert. Darüber hinaus wird die lokale Benutzergruppe "Siemens TIA Openness" erzeugt.

Hinweis

Sie haben mit dem Add-on-Paket "TIA Portal Openness" weiterhin keinen Zugriff auf das TIA Portal. Sie müssen Mitglied der Benutzergruppe "Siemens TIA Openness" sein (siehe Benutzer der Benutzergruppe "Siemens TIA Openness" hinzufügen (Seite 41)).

4.1.3 Benutzer der Benutzergruppe "Siemens TIA Openness" hinzufügen

Einleitung

Wenn Sie TIA Portal Openness auf dem PC installieren, wird die Benutzergruppe "Siemens TIA Openness" automatisch erzeugt.

Bei jedem Zugriff auf das TIA Portal mit Ihrer TIA Portal Openness-Anwendung prüft das TIA Portal, ob Sie Mitglied der Benutzergruppe "Siemens TIA Openness" sind, entweder direkt oder indirekt über eine andere Benutzergruppe. Wenn Sie Mitglied der Benutzergruppe "Siemens TIA Openness" sind, startet die TIA Portal Openness-Anwendung und stellt eine Verbindung zum TIA Portal her.

Vorgehensweise

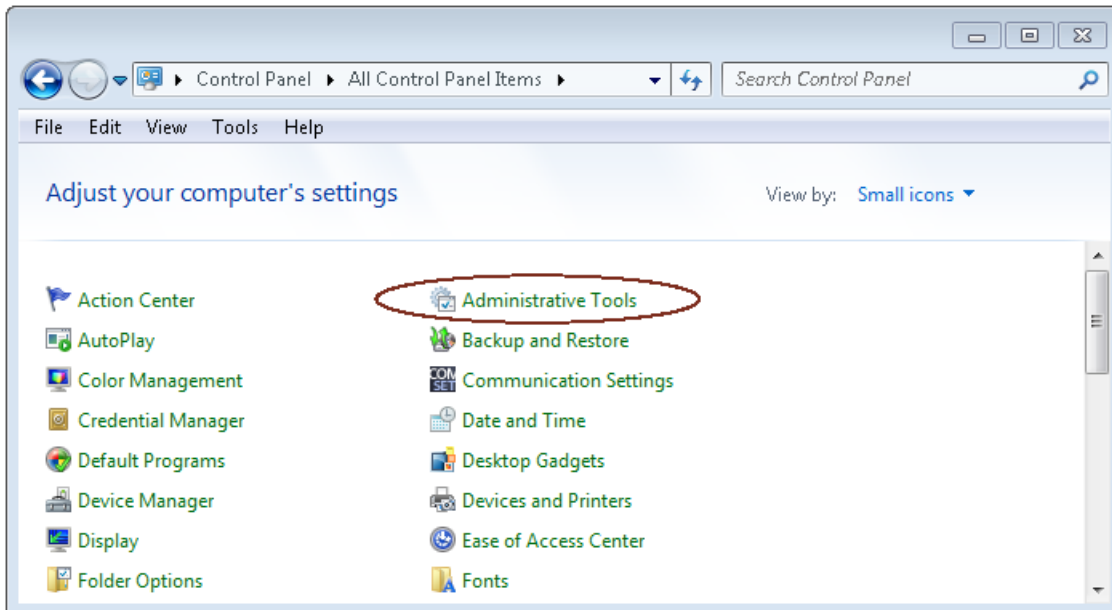
Wenn Sie der Benutzergruppe "Siemens TIA Openness" einen Benutzer hinzufügen möchten, müssen Sie auf Anwendungen aus Ihrem Betriebssystem zurückgreifen. Das TIA Portal unterstützt diesen Vorgang nicht.

Hinweis

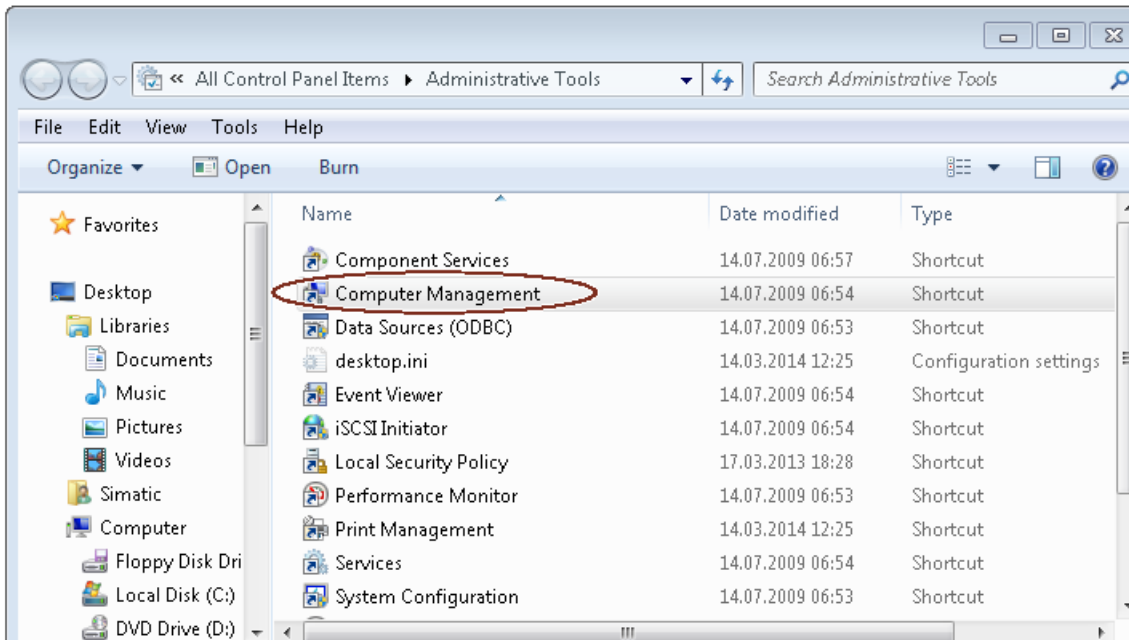
Je nach Konfiguration Ihrer Domain oder Ihres Rechners müssen Sie sich möglicherweise mit Administratorrechten anmelden, um die Benutzergruppe zu erweitern.

Im Betriebssystem Windows 7 (Spracheinstellung Englisch) gehen Sie zum Beispiel wie folgt vor, um einer Benutzergruppe einen Benutzer hinzuzufügen:

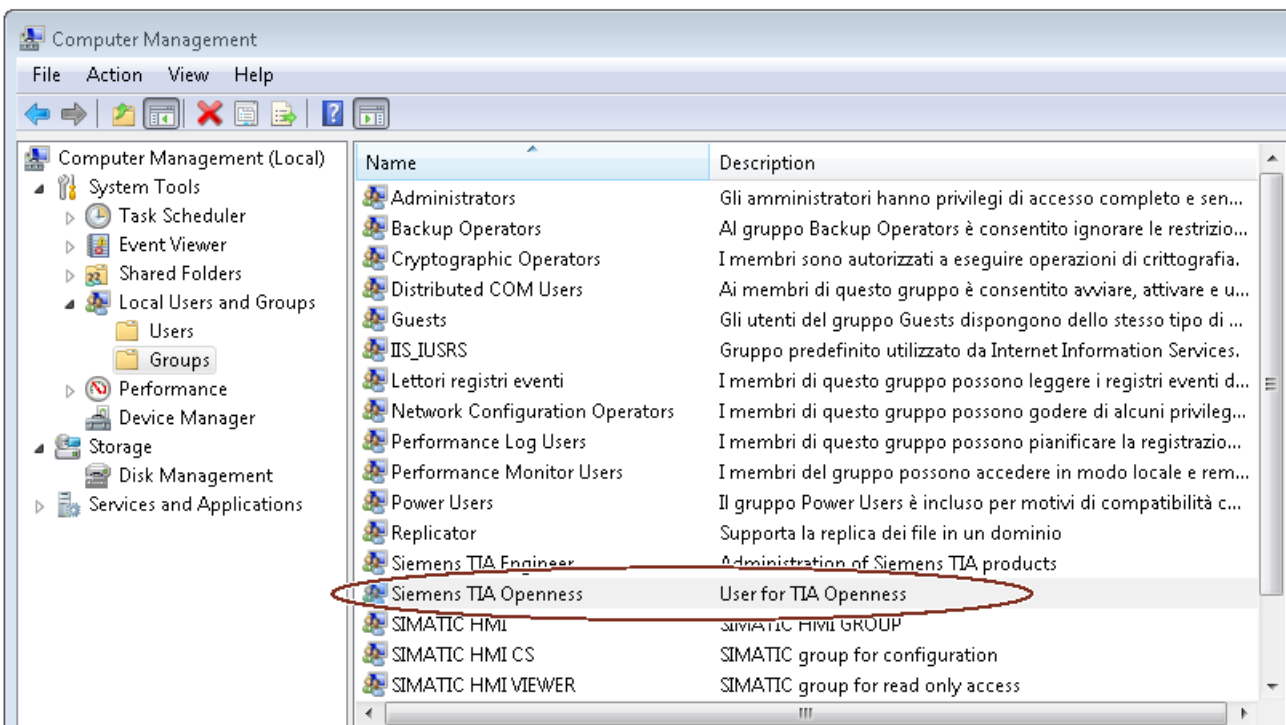
1. Wählen Sie "Start" > "Control Panel" aus.
2. Doppelklicken Sie im Control Panel auf "Administrative Tools".



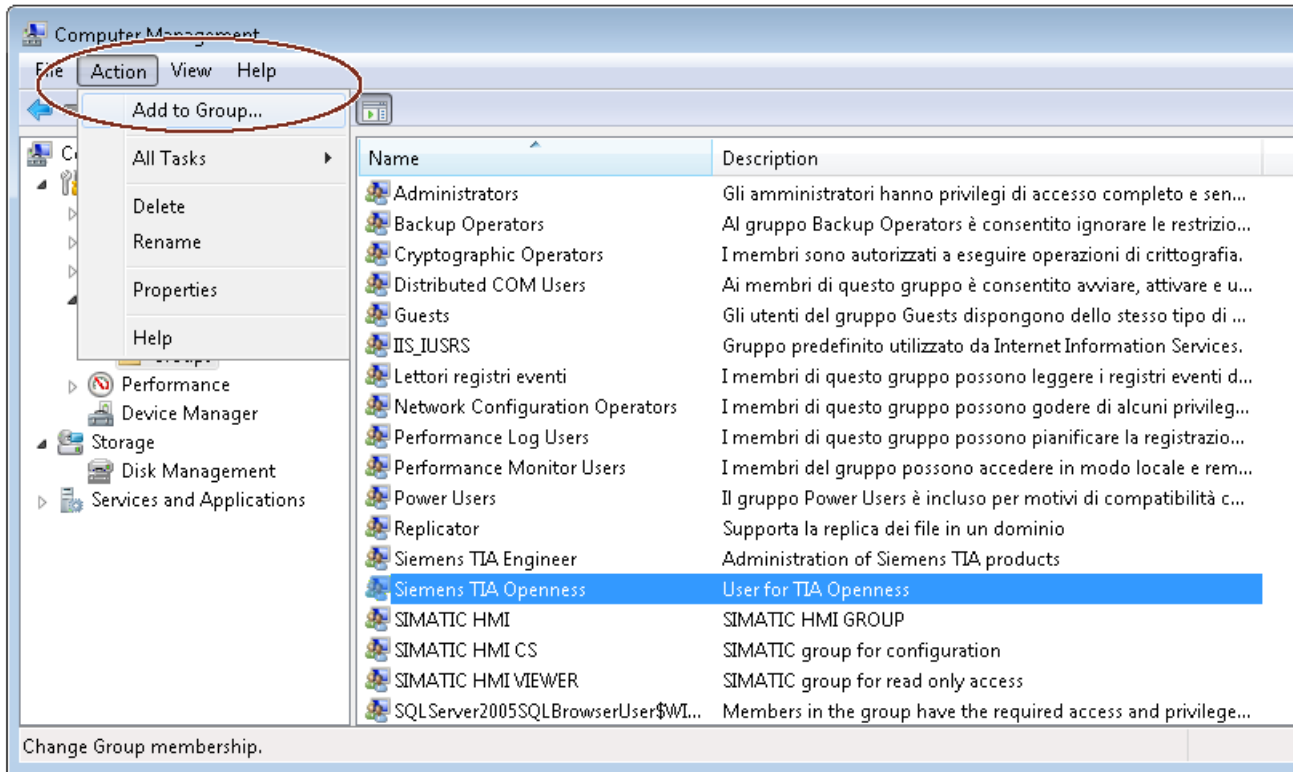
3. Klicken Sie auf "Computer Management", um den gleichnamigen Konfigurationsdialog zu öffnen.



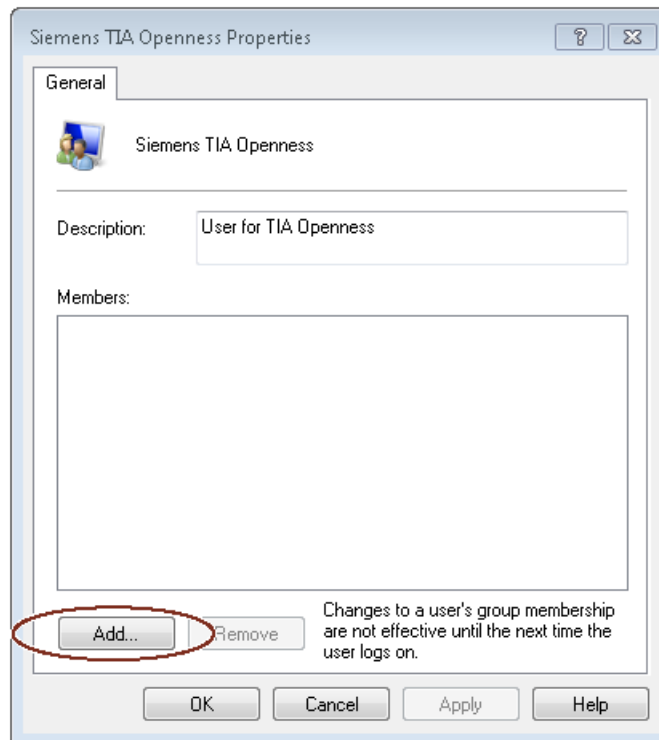
4. Wählen Sie "Local Users and Groups > Groups" aus, um alle erzeugten Benutzergruppen anzuzeigen.
5. Markieren Sie den Eintrag "Siemens TIA Openness" in der Liste der Benutzergruppen im rechten Fenster.



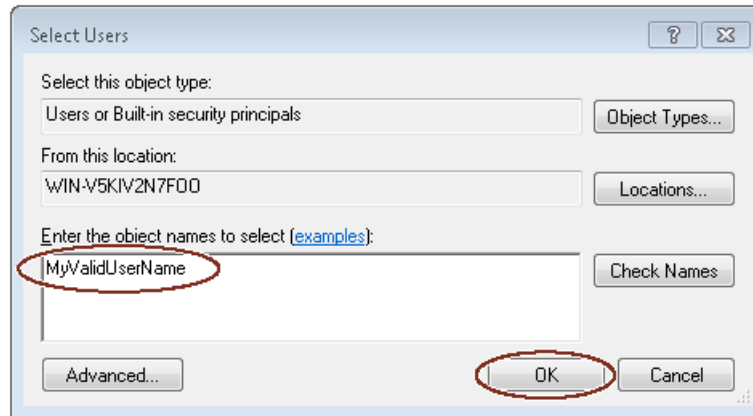
6. Wählen Sie den Menübefehl "Action > Add to Group..." aus.



Der Attributedialog der Benutzergruppe öffnet sich:



7. Klicken Sie auf "Add".
Der sich daraufhin öffnende Auswahldialog zeigt die auswählbaren Benutzer an:



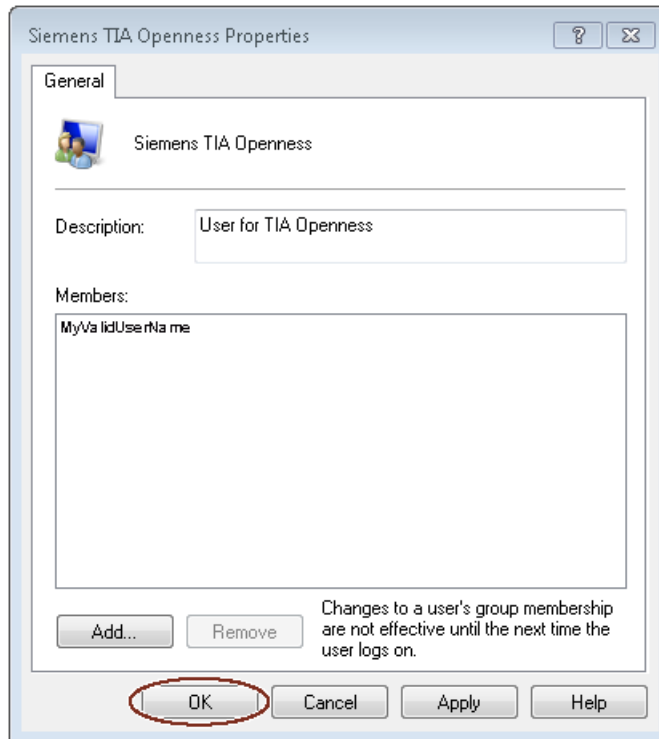
8. Geben Sie einen gültigen Benutzernamen in das Eingabefeld ein.

Hinweis

Klicken Sie auf "Check Names", um zu prüfen, ob der Benutzer ein gültiges Benutzerkonto für diese Domain oder diesen Rechner hat.

Im Feld "From this location" wird der Name der Domain oder des Rechners für den eingegebenen Benutzernamen angezeigt. Weitere Informationen hierzu erhalten Sie von Ihrem Systemadministrator.

9. Bestätigen Sie Ihre Auswahl mit "OK".
Der neue Benutzer wird jetzt im Attributedialog der Benutzergruppe angezeigt.



Zusätzliche Benutzer registrieren Sie durch Klicken auf die Schaltfläche "Add".

10. Klicken Sie auf "OK", um diesen Vorgang abzuschließen.
11. Melden Sie sich neu beim PC an, damit die Änderungen wirksam werden.

4.1.4 Verwendung von Windows-Diensten

Einleitung

Windows-Dienste sind ausführbare Dateien mit langer Ausführungsdauer und ohne Benutzeroberfläche, die mithilfe eines Installationsprogramms installiert werden. Es ist möglich, eine solche Windows-Dienstanwendung zu entwickeln, die TIA Portal Openness nutzt.

Dazu muss in Microsoft Visual Studio ein Projekt vom Typ "Windows-Dienst (.NET Framework)" erstellt werden.

Voraussetzung

- Die ausführbare Prozessdatei der Windows-Dienstanwendung muss vorher der Whitelist der TIA Portal Openness-Firewall hinzugefügt werden. Dies kann durch Erstellen der Einträge in der Windows-Registrierung bei der Installation des Windows-Diensts erfolgen, beispielsweise als Teil eines mit einem Windows Installer-Projekt erstellten Setups. Weitere Informationen zum Hinzufügen von Einträgen zur Whitelist finden Sie im Abschnitt "Whitelist-Eintrag ohne TIA Portal hinzufügen" im Kapitel TIA Portal Openness-Firewall (Seite 90)
- Der Zugriff auf aktive TIA Portal-Prozesse bzw. das Starten von TIA Portal über die grafische Benutzeroberfläche ist nicht möglich. Es kann lediglich eine neue TIA Portal-Instanz ohne Benutzeroberfläche erstellt werden.
- Es wird dringend empfohlen, den Windows-Dienst mit den Anmeldeinformationen eines eigenen Windows-Benutzerkontos auszuführen. Die Verwendung des lokalen Dienstkontos oder des Netzwerkdienstkontos ist zwar möglich, wird jedoch nicht empfohlen. Das lokale Systemkonto wird aufgrund von Sicherheitseinschränkungen nicht unterstützt und löst immer eine EngineeringSecurityException aus.
- Das verwendete Windows-Konto (entweder Dienst oder Benutzer) muss Teil der lokalen Windows-Benutzergruppe "Siemens TIA Openness" sein.

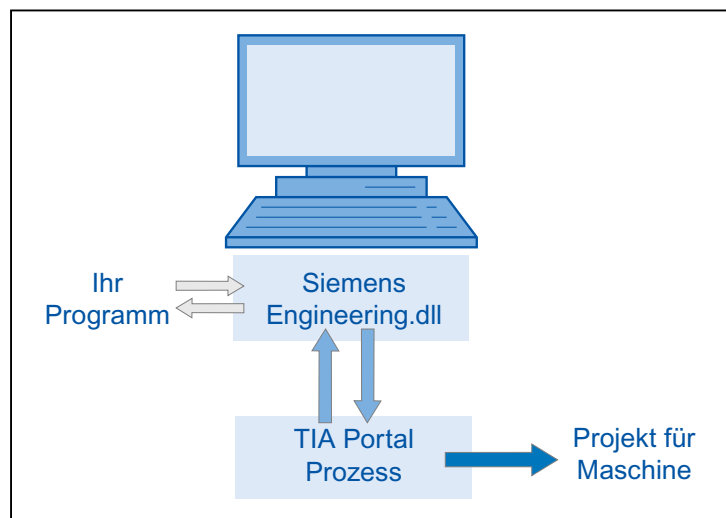
Siehe auch

TIA Portal Openness-Firewall (Seite 90)

4.1.5 Auf das TIA Portal zugreifen

Übersicht

Projektierungsrechner



Vorgehensweise

1. Um auf das TIA Portal zuzugreifen und zu starten, richten Sie die Entwicklungsumgebung ein.
2. Um das Portal zu starten, instanziiieren Sie in Ihrem Programm das Objekt der Portalanwendung.
3. Suchen Sie das gewünschte Projekt und öffnen Sie es
4. Greifen Sie auf die Projektdaten zu.
5. Schließen Sie das Projekt und beenden Sie das TIA Portal.

Siehe auch

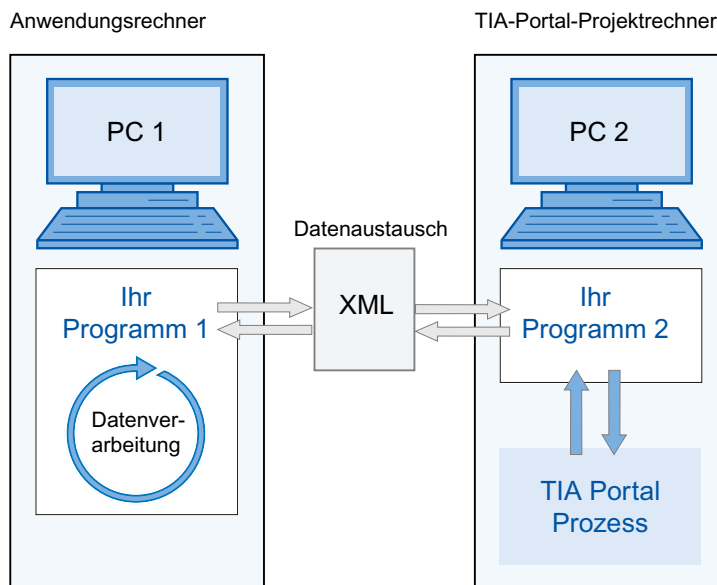
Verbindung zum TIA Portal aufbauen (Seite 84)

Verbindung zum TIA Portal beenden (Seite 101)

4.1.6 Konfigurationen

Sie können mit zwei Varianten von TIA Portal Openness arbeiten:

Die Anwendung und das TIA Portal befinden sich auf unterschiedlichen Computern



- Der Datenaustausch erfolgt über XML-Dateien. Die XML-Dateien können von Ihren Programmen exportiert oder importiert werden.
- Die aus dem TIA Portal-Projekt auf PC2 exportierten Daten können auf PC1 geändert und wieder importiert werden.

Hinweis

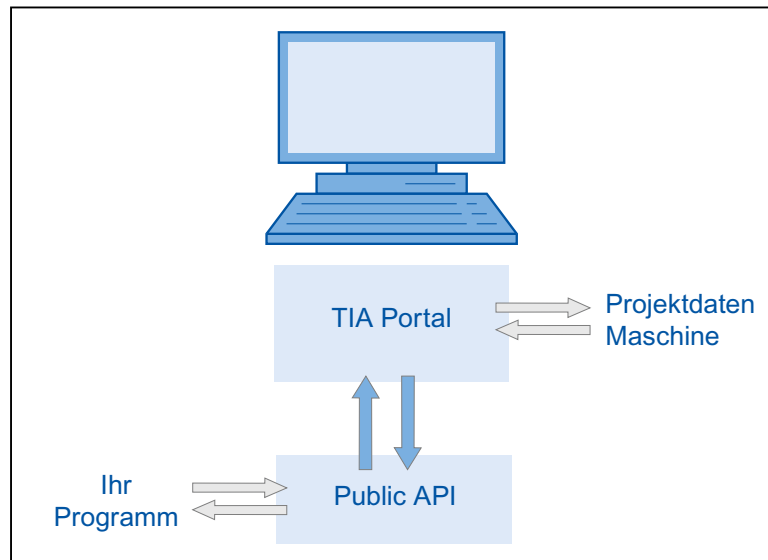
Sie müssen für PC2 ein ausführbares Programm „Ihr Programm 2“ entwickeln, beispielsweise „programm2.exe“. Das TIA Portal läuft mit diesem Programm im Hintergrund.

Import und Export der XML-Dateien erfolgt ausschließlich über die TIA Portal Openness API.

- Sie können ausgetauschte Dateien zu Verifizierungszwecken archivieren.
- Ausgetauschte Daten können an unterschiedlichen Standorten und zu unterschiedlichen Zeiten verarbeitet werden.

Die Anwendung und das TIA Portal befinden sich auf demselben Computer

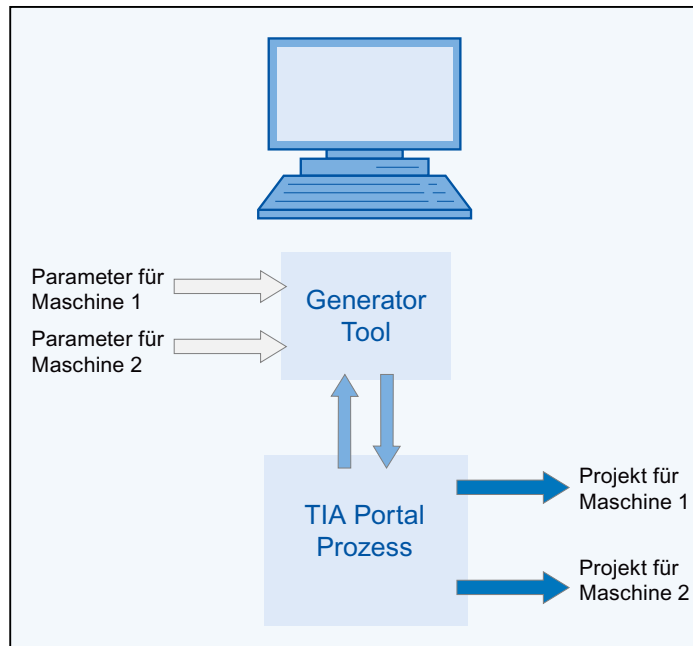
Projektierungsrechner



- Ihr Programm ruft das TIA Portal entweder mit der oder ohne die Benutzeroberfläche auf. Ihr Programm öffnet, speichert und/oder schließt ein Projekt. Das Programm kann auch eine Verbindung zu einem laufenden TIA Portal herstellen.
- Sie können dann die TIA Portal-Funktionalität nutzen, um Projektdaten anzufordern, zu generieren und zu ändern oder um Import- und Exportvorgänge auszulösen.
- Die Daten werden unter der Kontrolle der TIA Portal-Verarbeitung erstellt und in den Projektdaten gespeichert.

Typischer Anwendungsbereich ist der modulare Maschinenbau

Projektierungsrechner



- Ein effizientes Automatisierungssystem ist auf ähnliche Maschinen anzuwenden.
- Ein Projekt ist im TIA Portal verfügbar, das die Komponenten aller Maschinenvarianten enthält.
- Das Generator-Tool steuert die Erstellung des Projekts für eine bestimmte Maschinenvariante.
- Das Generator-Tool ruft die Standardwerte ab, indem es die Parameter für die angeforderte Maschinenvariante einliest.
- Das Generator-Tool filtert die betreffenden Elemente aus dem TIA Portal-Gesamtprojekt heraus, ändert sie gegebenenfalls und generiert das angeforderte Maschinenprojekt.

4.2 Aufgaben von Openness

4.2.1 Einführung

Einleitung

In TIA Portal Openness werden offene Schnittstellen für das Engineering mit dem TIA Portal beschrieben. Weitere Informationen über "TIA Portal Openness - Efficient generation of program code using code generators" finden Sie im YouTube-Kanal von SIEMENS (www.youtube.com/watch?v=Ki12pLbEcxs).

Sie automatisieren das Engineering mit TIA Portal Openness, indem Sie das TIA Portal extern über ein von Ihnen erstelltes Programm steuern.

Mit TIA Portal Openness können Sie folgende Aktionen durchführen:

- Projektdaten erstellen
- Projekte und Projektdaten ändern
- Projektdaten löschen
- Projektdaten einlesen
- Projekte und Projektdaten anderen Anwendungen zur Verfügung stellen

Hinweis

Siemens ist nicht verantwortlich für die Kompatibilität von Daten und Informationen, die über diese Schnittstellen mit Fremdsoftware übertragen werden, und übernimmt dafür keine Gewähr.

Wir weisen ausdrücklich darauf hin, dass eine unsachgemäße Nutzung der Schnittstellen zu Datenverlust und zum Produktionsstillstand führen kann.

Hinweis

Die in dieser Dokumentation enthaltenen Codeausschnitte sind in C#-Syntax geschrieben.

Aufgrund der Kürze der verwendeten Codeausschnitte hat sich die Beschreibung der Fehlerbehandlung ebenfalls verkürzt.

Verwendung

Die TIA Portal Openness-Schnittstelle dient folgenden Zwecken:

- Projektdaten bereitstellen
- Auf den TIA Portal-Prozess zugreifen
- Projektdaten nutzen

Verwendung von Standardwerten aus dem Bereich der Automatisierungstechnik

- Durch Importieren von extern generierten Daten
- Durch Fernsteuerung des TIA Portals zum Generieren von Projekten

Bereitstellen von Projektdaten des TIA Portals für externe Anwendungen

- Durch Exportieren von Projektdaten

Sichern von Wettbewerbsvorteilen durch effizientes Engineering

- Sie brauchen vorhandene Engineering-Daten im TIA Portal nicht zu konfigurieren.
- Automatisierte Engineering-Prozesse ersetzen das manuelle Engineering.
- Niedrige Engineering-Kosten stärken die Anbieterposition gegenüber Mitbewerbern.

Gemeinsam an Projektdaten arbeiten

- Testroutinen und Massendatenverarbeitung können parallel zur laufenden Konfiguration ablaufen bzw. stattfinden.

Siehe auch

Konfigurationen (Seite 48)

4.2.2 Einsatzmöglichkeiten

Einleitung

TIA Portal Openness bietet Ihnen verschiedene Möglichkeiten für den Zugriff auf das TIA Portal und eine Auswahl von Funktionen für definierte Aufgaben.

Über die Schnittstelle TIA Portal Openness API greifen Sie auf folgende Bereiche des TIA Portals zu:

- Portaldaten
- Projektdaten
- PLC-Daten
- HMI-Daten
- Antriebsdaten

Auf das TIA Portal zugreifen

TIA Portal Openness bietet verschiedene Möglichkeiten für den Zugriff auf das TIA Portal. Sie erzeugen eine externe Instanz des TIA Portals im Prozess mit oder ohne Benutzeroberfläche. Sie können auch gleichzeitig auf laufende Prozesse des TIA Portals zugreifen.

Auf Projekte und Projektdaten zugreifen

Beim Zugriff auf Projekte und Projektdaten verwenden Sie TIA Portal Openness hauptsächlich für die folgenden Aufgaben:

- Projekt öffnen, schließen und speichern
- Objekte enumerieren und abfragen
- Objekte erzeugen
- Objekte löschen

4.2.3 Export/Import

Einleitung

TIA Portal Openness unterstützt den Import und Export von Projektdaten mittels XML-Dateien. Die Funktion Importieren/Exportieren unterstützt die externe Konfiguration vorhandener Engineering-Daten. Sie verwenden diese Funktion, um den Engineering-Prozess effektiv und fehlerfrei zu machen.

Verwendung

Sie nutzen die Funktion Importieren/Exportieren für die folgenden Zwecke:

- Datenaustausch
- Kopieren von Teilen eines Projekts
- Externe Verarbeitung von Konfigurationsdaten, zum Beispiel für Massendatenoperationen mit Suchen und Ersetzen
- Externe Verarbeitung von Konfigurationsdaten für neue Projekte auf der Grundlage bestehender Konfigurationen
- Importieren extern erzeugter Konfigurationsdaten, zum Beispiel Textlisten und Variablen
- Bereitstellen von Projektdaten für externe Anwendungen

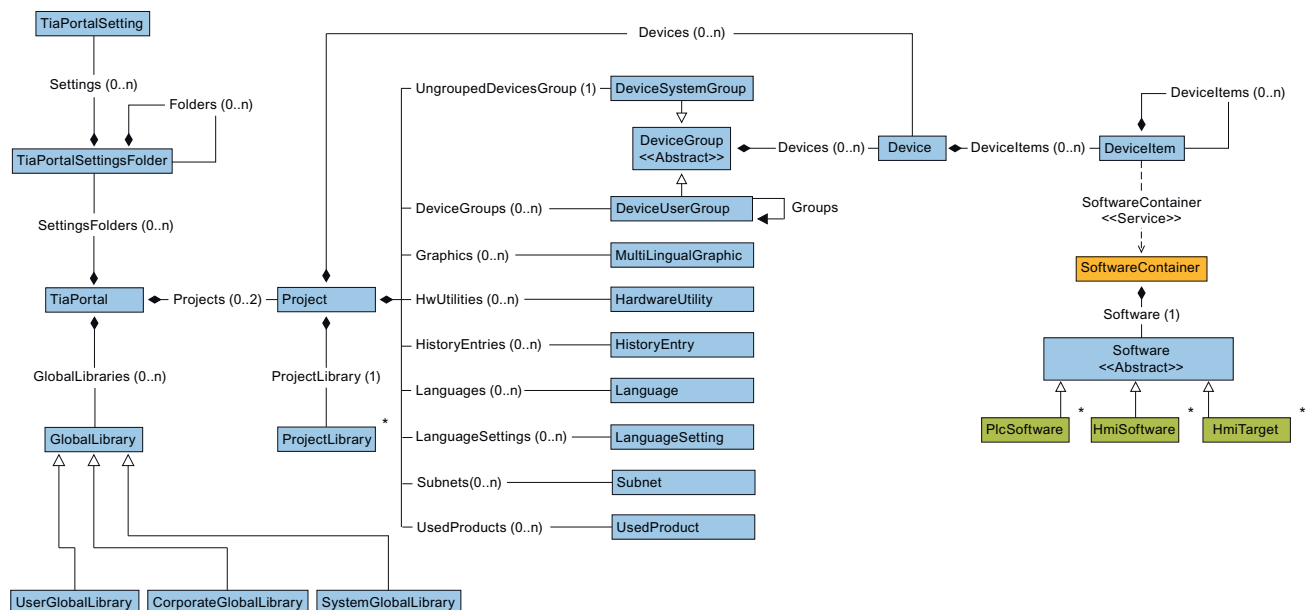
TIA Portal Openness API

5.1 TIA Portal Openness-Objekt

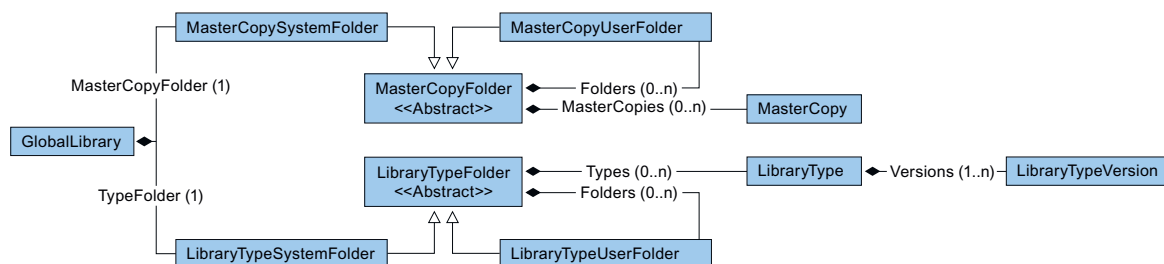
5.1.1 TIA Portal Openness-Objektmodell

Übersicht

Das folgende Diagramm beschreibt die höchste Ebene des TIA Portal Openness-Objektmodells:

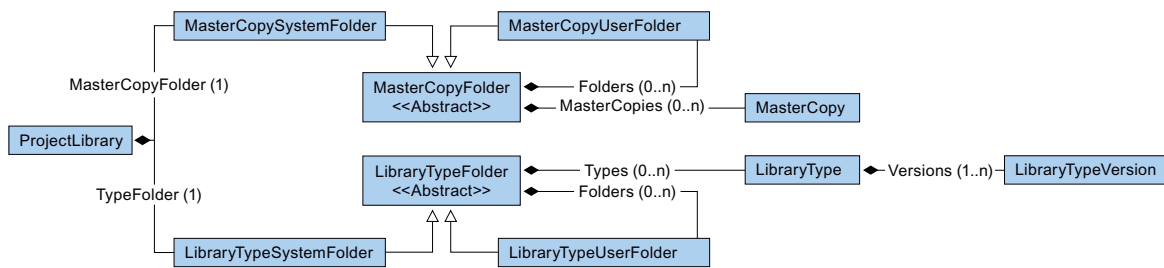


Das folgende Diagramm beschreibt die Objekte, die sich unter GlobalLibrary befinden.

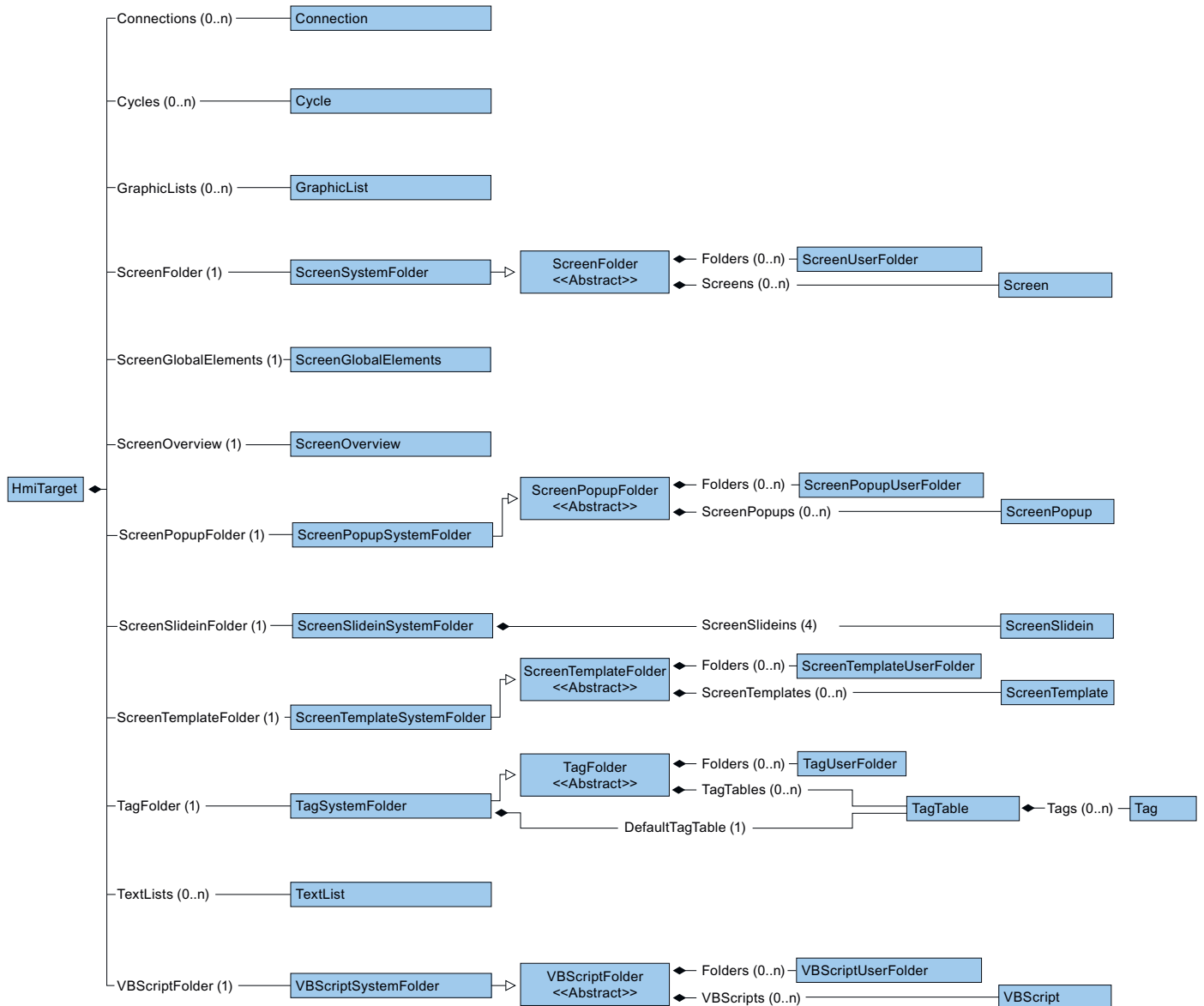


Das folgende Diagramm beschreibt die Objekte, die sich unter ProjectLibrary befinden.

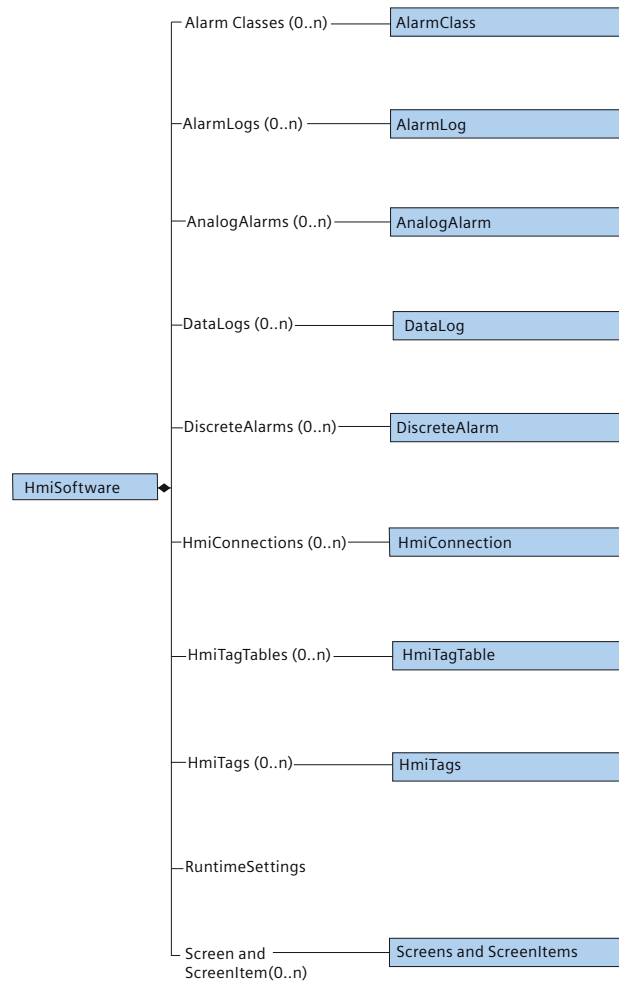
5.1 TIA Portal Openness-Objekt



Das folgende Diagramm beschreibt die Objekte, die sich unter HmiTarget befinden.



Das folgende Diagramm beschreibt die Objekte, die sich unter HmiSoftware befinden.



Das folgende Diagramm beschreibt die Objekte, die sich unter PlcSoftware befinden.



* Die Forcetabelle muss sich in PlcWatch und ForceTableSystemGroup befinden.

Auf Objekte in Listen zugreifen

Sie haben die folgenden Optionen zum Adressieren eines Objekts in einer Liste:

- Adressieren Sie über den Index. Die Zählung innerhalb der Listen beginnt mit 0.
- Verwenden Sie die Methode `Find`.
Verwenden Sie diese Methode zum Adressieren eines Objekts über seinen Namen. Sie können diese Methode für eine Zusammensetzung oder Liste verwenden. Die Methode `Find` ist nicht rekursiv.

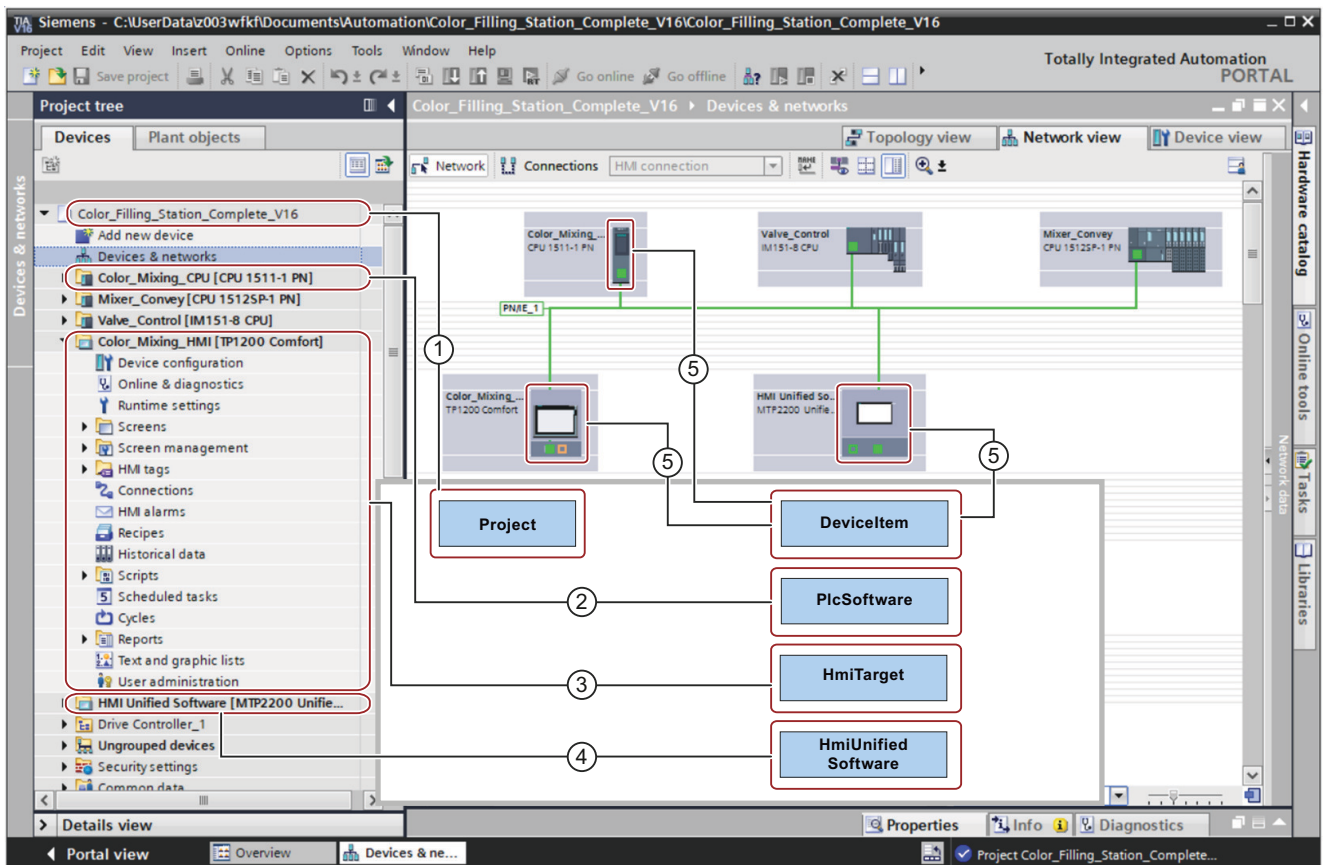
Beispiel:

```
ScreenComposition screens = folder.Screens;  
Screen screen = screens.Find("myScreen");
```

- Verwenden Sie symbolische Namen.

Beziehung zwischen TIA Portal und TIA Portal Openness-Objektmodell

Das Bild unten zeigt die Beziehung zwischen dem Objektmodell und einem Projekt im TIA Portal:



Siehe auch

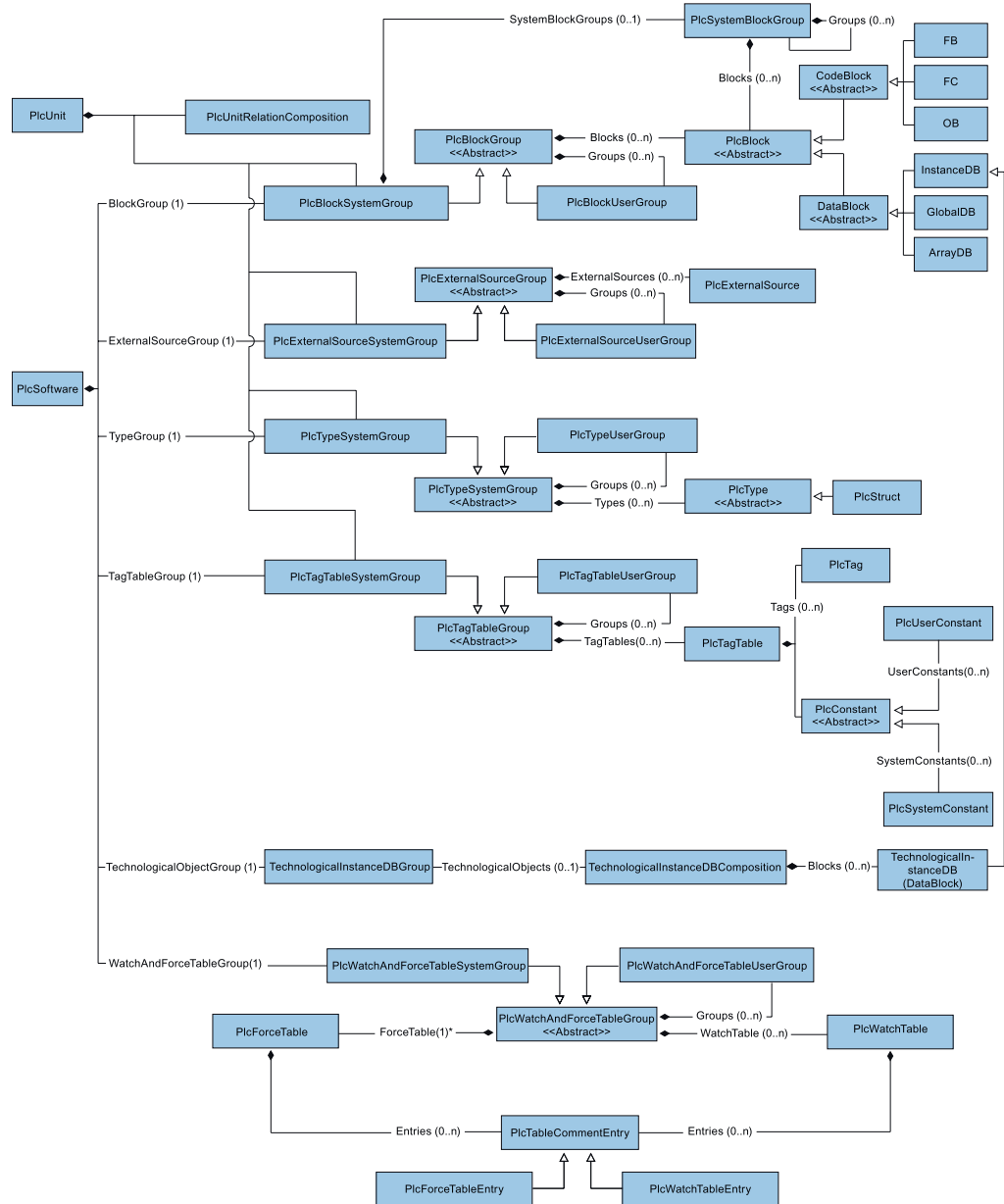
Bausteine und Typen des TIA Portal Openness-Objektmodells (Seite 61)

Hierarchie von Hardware-Objekten des Objektmodells (Seite 67)

5.1.2 Bausteine und Typen des TIA Portal Openness-Objektmodells

Einleitung

Das folgende Diagramm beschreibt das Domänenmodell der PLCs, um einen Überblick über die aktuelle Modellierung in TIA Portal Openness zu geben.



Hinweis

* Die Forcetabelle muss sich in PlcWatch und ForceTableSystemGroup befinden

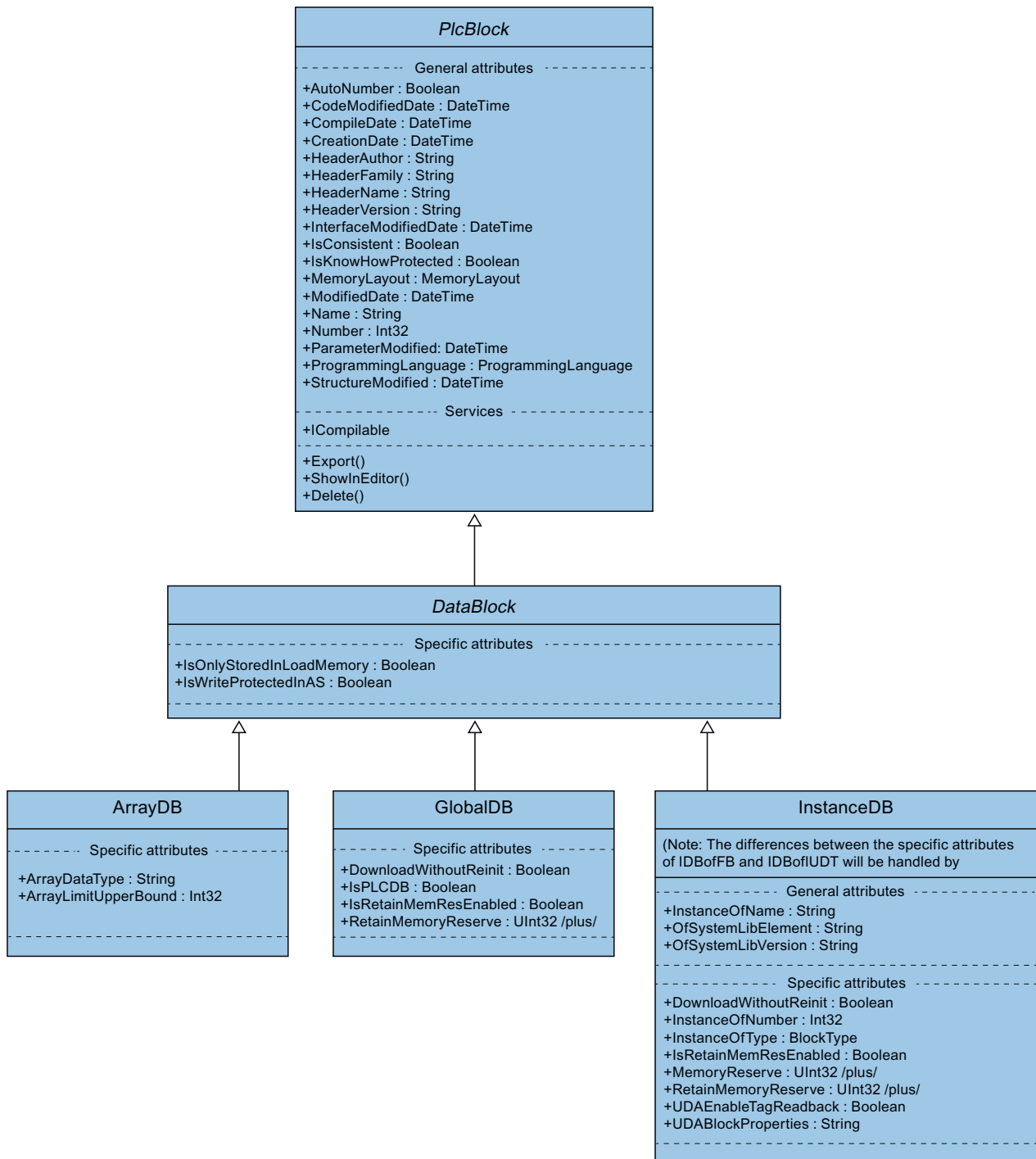
Darstellung von Bausteinen und Typen in der TIA Portal Openness API

Der vereinfachte Modellteil von Bausteinen und für die Struktur basiert auf den Attributen in der TIA Portal Openness API. Diese Klassen bieten die Exportfunktion und für Bausteine auch die Übersetzungsfunktion.

Klassendiagramme

Im TIA Portal Openness-Objektmodell sind alle nicht direkt instanziierten Klassen als abstrakt definiert.

Daten



5.1 TIA Portal Openness-Objekt

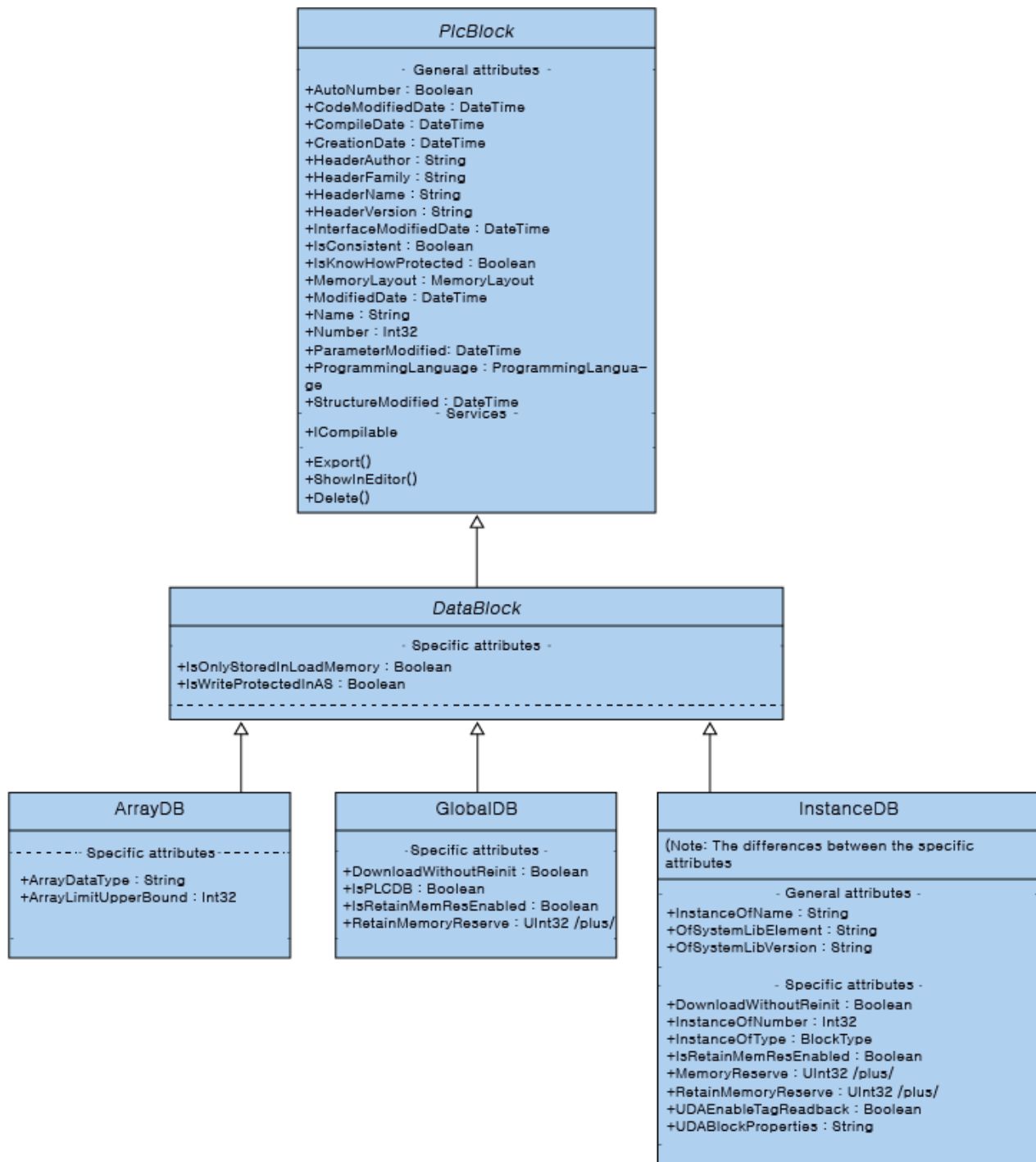
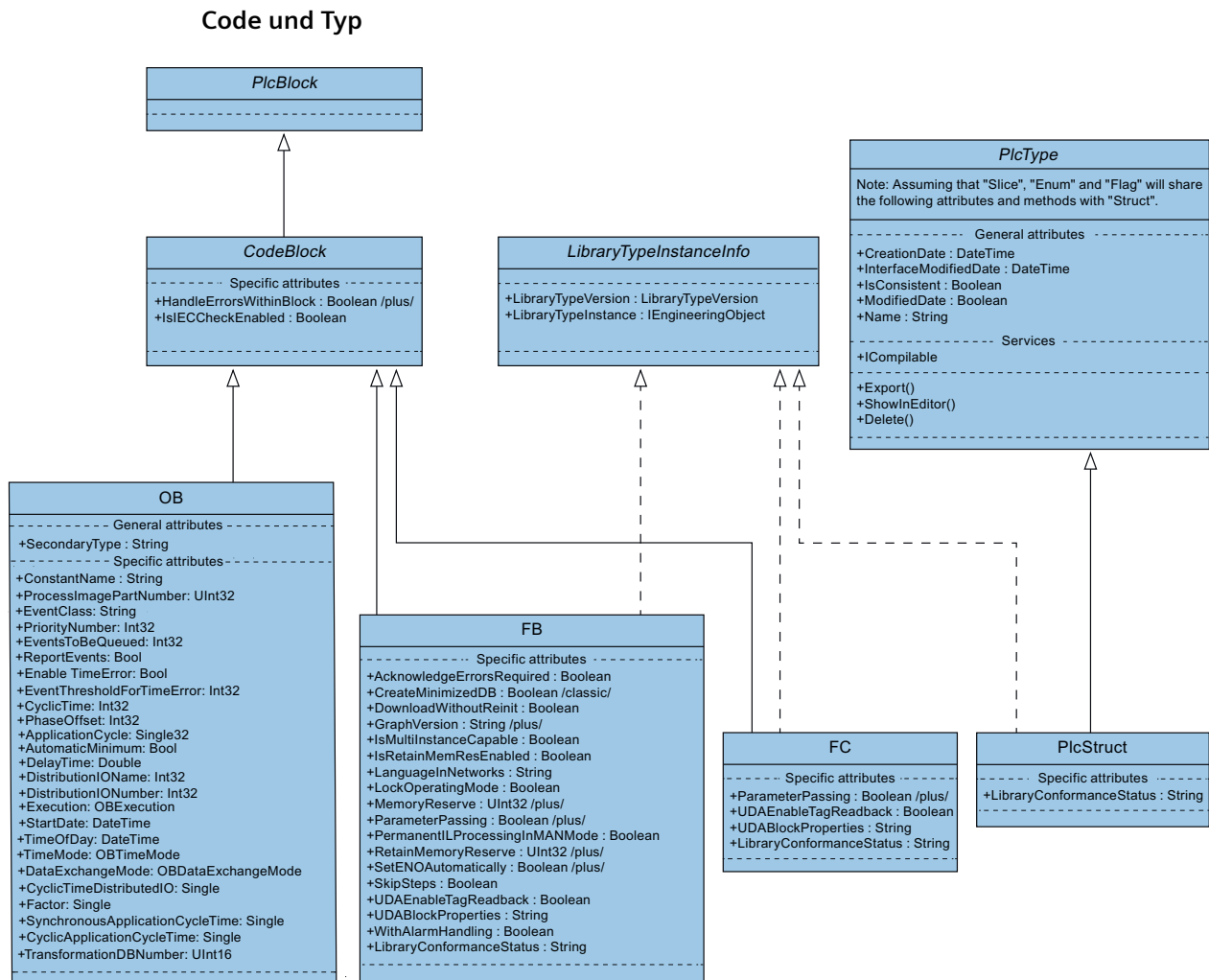


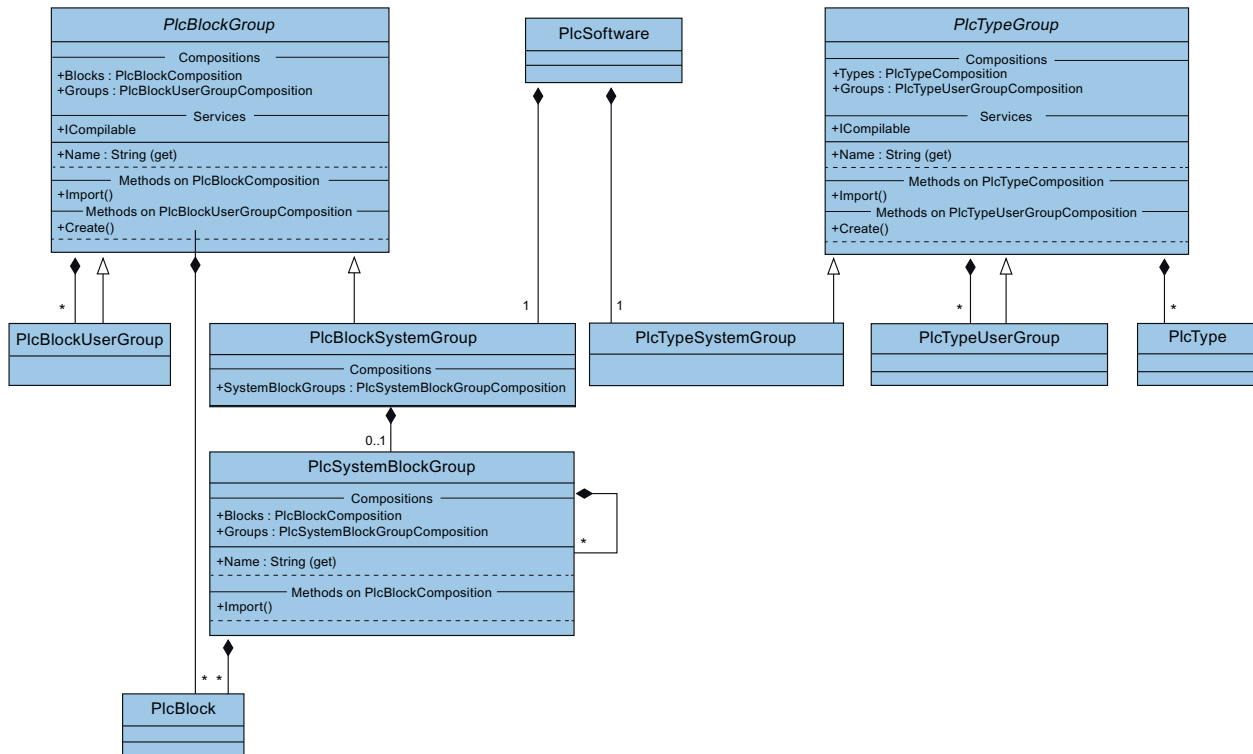
Bild 5-1 Darstellung eines Blockschaltbilds



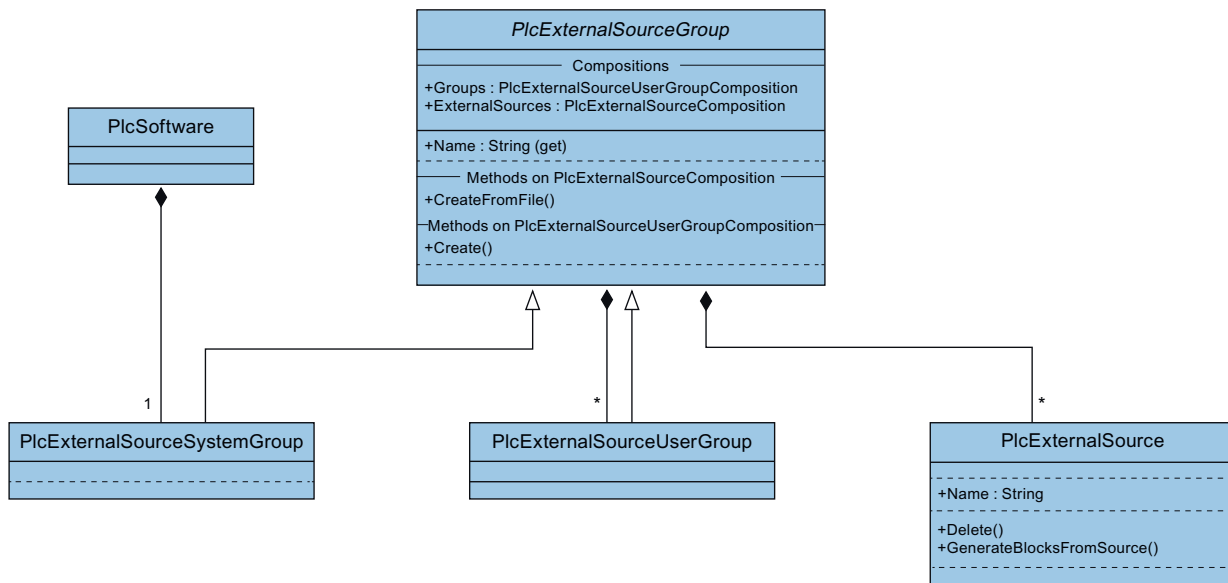
Darstellung von Gruppen für Bausteine und Typen in der TIA Portal Openness API

Die zwei Gruppen "PlcBlocks" der oberen Ebene ("Programmbausteine" in der Benutzeroberfläche des TIA Portals) und "PlcTypes" ("PLC-Datentypen" in der Benutzeroberfläche des TIA Portals) enthalten Bausteine und Typdefinitionen. Diese Gruppen stellen die Import- und die Übersetzungsfunktion für Bausteine zur Verfügung. Aufgrund der Tatsache, dass die meisten Methoden von Funktionalitäten der Gruppen nur über Sammlungen erreichbar sind, gibt es eine "eingebettete" oder "verdichtete" Darstellung der Sammlungen und ihrer Methoden in den "Host"-Klassen.

Bausteine und Typen



Externe Quellen



Siehe auch

TIA Portal Openness-Objektmodell (Seite 55)

Hierarchie von Hardware-Objekten des Objektmodells (Seite 67)

5.1.3 Hierarchie von Hardware-Objekten des Objektmodells

Beziehung zwischen den sichtbaren Elementen im TIA Portal und den modellierten Elementen im Objektmodell

Hardware-Objekt	Erläuterung
Gerät (Device)	Das Behälterobjekt für eine zentrale oder dezentrale Konfiguration
Geräteelement (DeviceItem)	Jedes Geräteelementobjekt hat ein Behälterobjekt. Die logische Beziehung ist "Elemente".

Um die Hierarchie von Hardware-Objekten bezüglich der Beziehung zwischen den in der TIA Portal-Ansicht angezeigten und den in TIA Portal Openness modellierten Elementen zu verstehen, müssen einige Aspekte der Hierarchie von Hardware-Objekten berücksichtigt werden.

Beispiel: Ein HardwareObject enthält die items-Verknüpfung. Das Device erweitert das HardwareObject und enthält die UnpluggedItems-Verknüpfung. Das Device ist das Behälterobjekt für eine Station.

Das DeviceItem erweitert das HardwareObject und enthält die Container-Verknüpfung. Jedes gesteckte DeviceItem-Objekt verfügt über ein Behälterobjekt, das ein HardwareObject darstellt. Die umgekehrte Beziehung ist Items.

Ein nicht gestecktes DeviceItem-Objekt ohne Behälter ist über die Verknüpfung UnpluggedItems des Geräts auffindbar.

Die Behälterbeziehung ist vergleichbar mit der Beziehung der Module für die Geräteelementobjekte.

Beispiel: Ein Gerät umfasst einen oder mehrere Baugruppenträger. Ein Baugruppenträger umfasst Module. Ein Modul umfasst Submodule.

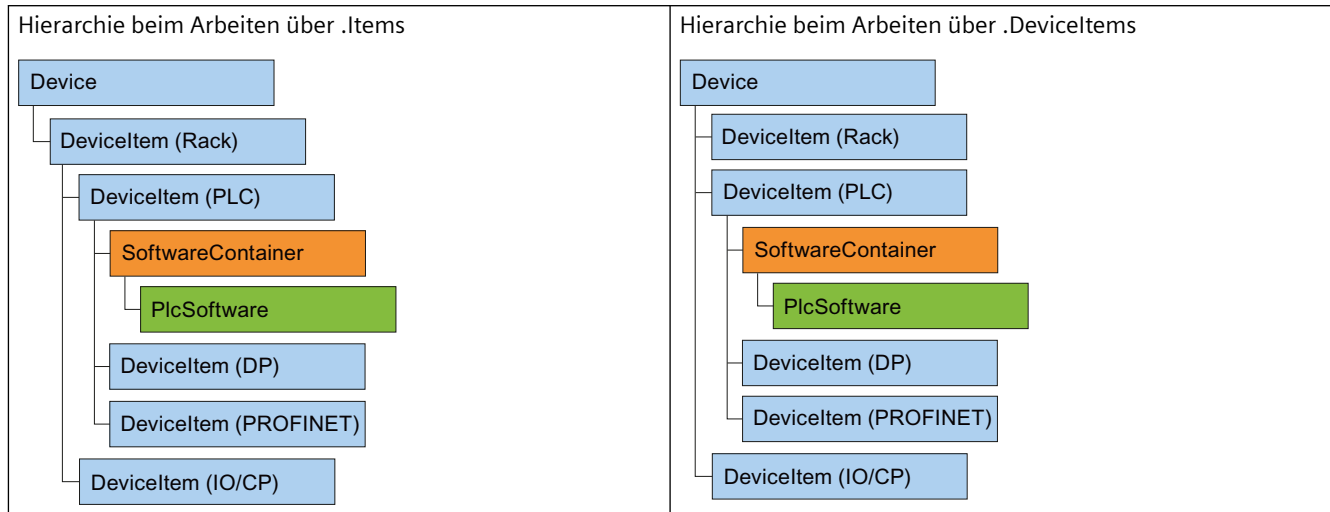
Hierbei handelt es sich um die Beziehung, die der Darstellung in Netzsicht und Gerätesicht des TIA Portals ähnelt. Das Attribut "PositionNumber" eines Geräteelements ist im Elementebereich innerhalb eines Behälters eindeutig.

Die Eltern-Kind-Beziehung zwischen Geräteelementobjekten ist eine rein logische Beziehung im Objektmodell. Ein Kind kann nicht ohne seine Eltern existieren.

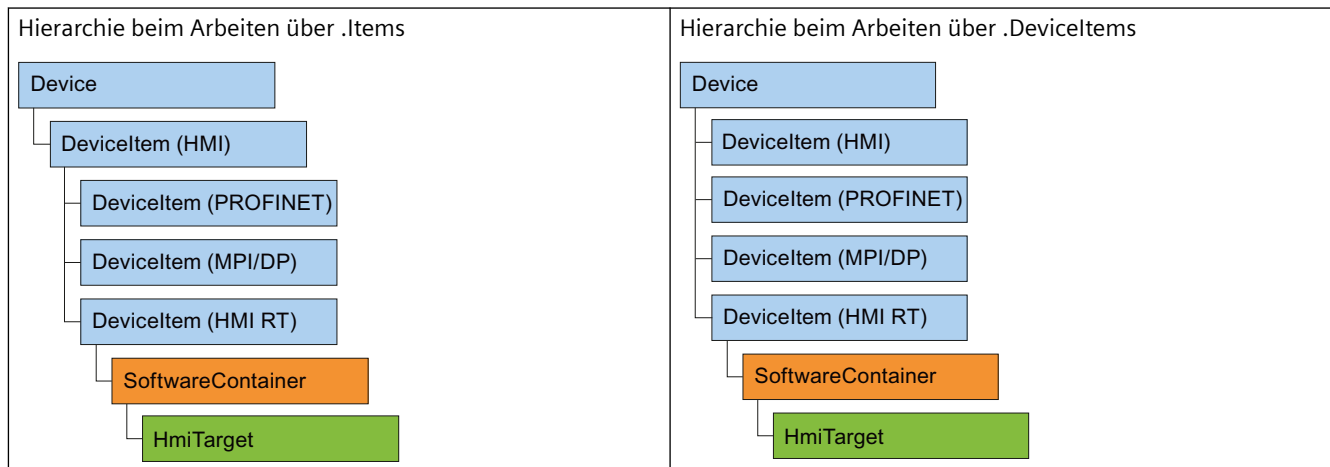
- Wenn ein Submodul als Teil eines Moduls modelliert wird (als Kind), kann das Submodul nicht ohne das Modul entfernt werden.
- Ein Modul oder Submodul, das hinzugefügt und entfernt werden kann, ist ein Kind des Geräteobjekts.

Hierarchie zwischen PLC-Geräten

Das Diagramm unten zeigt die Hierarchie zwischen Geräten und Geräteelementen von PLC- und HMI-Geräten.



Hierarchie zwischen HMI-Geräten



Siehe auch

TIA Portal Openness-Objektmodell (Seite 55)

Bausteine und Typen des TIA Portal Openness-Objektmodells (Seite 61)

5.1.4 Objektliste

Einleitung

In den folgenden Tabellen werden die verfügbaren Objekte bis einschließlich Runtime Advanced aufgeführt und es wird angegeben, ob diese Objekte von TIA Portal Openness unterstützt werden.

TIA Portal Openness in WinCC unterstützt weder die Visualisierungssoftware Runtime Professional noch Geräteproxydateien.

Objekte

Je nach dem von Ihnen genutzten Bediengerät können Sie die folgenden Projektdaten verwenden:

Tabelle 5-1 Bilder

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Bild	Ja	Ja	Ja	Ja
Globales Bild	Ja	Ja	Ja	Ja
Vorlagen	Ja	Ja	Ja	Ja
Permanentfenster	Nein	Ja	Ja	Ja
Pop-up-Bild	Nein	Ja	Ja	Ja
Slide-in-Bild	Nein	Ja	Ja	Ja

Tabelle 5-2 Bildobjekte

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Linie	Ja	Ja	Ja	Ja
Polygonzug	Nein	Ja	Ja	Ja
Polygon	Nein	Ja	Ja	Ja
Ellipse	Ja	Ja	Ja	Ja
Ellipsensegment	Nein	Nein	Nein	Nein
Kreissegment	Nein	Nein	Nein	Nein
Ellipsenbogen	Nein	Nein	Nein	Nein
Kamera-Anzeige	Nein	Nein	Nein	Nein
Kreisbogen	Nein	Nein	Nein	Nein
Kreis	Ja	Ja	Ja	Ja
PDF-Anzeige	Nein	Nein	Nein	Nein
Rechteck	Ja	Ja	Ja	Ja
Verbinder	Nein	Nein	Nein	Nein
Textfeld	Ja	Ja	Ja	Ja
Grafikanzeige	Ja	Ja	Ja	Ja
Rohr	Nein	Nein	Nein	Nein

5.1 TIA Portal Openness-Objekt

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Doppel-T-Stück	Nein	Nein	Nein	Nein
T-Stück	Nein	Nein	Nein	Nein
Rohrkrümmer	Nein	Nein	Nein	Nein
E/A-Feld	Ja	Ja	Ja	Ja
Datum/Uhrzeit-Feld	Ja	Ja	Ja	Ja
Grafisches E/A-Feld	Ja	Ja	Ja	Ja
Editierbares Textfeld	Nein	Nein	Nein	Nein
Listenfeld	Nein	Nein	Nein	Nein
Kombinationsfeld	Nein	Nein	Nein	Nein
Schaltfläche	Ja	Ja	Ja	Ja
Rundschaltfläche	Nein	Nein	Nein	Nein
Leuchtdrucktaster	Nein	Nein	Ja	Nein
Schalter	Ja	Ja	Ja	Ja
Symbolisches E/A-Feld	Ja	Ja	Ja	Ja
Schlüsselschalter	Nein	Nein	Ja	Nein
Balken	Ja	Ja	Ja	Ja
Symbolbibliothek	Nein	Ja	Ja	Ja
Schieberegler	Nein	Ja	Ja	Ja
Bildlaufleiste	Nein	Nein	Nein	Nein
Kontrollkästchen	Nein	Nein	Nein	Nein
Optionsschaltfläche	Nein	Nein	Nein	Nein
Zeigerinstrument	Nein	Ja	Ja	Ja
Uhr	Nein	Ja	Ja	Ja
Speicherplatzanzeige	Nein	Nein	Nein	Nein
Funktionstasten	Ja	Ja	Ja	Ja
Bildbausteininstanzen	Nein	Ja	Ja	Ja
Bildfenster	Nein	Nein	Nein	Nein
Benutzeranzeige	Ja	Ja	Ja	Ja
HTML-Browser	Nein	Nein	Nein	Nein
Druckauftrag/Skriptdiagnose	Nein	Nein	Nein	Nein
Rezepturanzeige	Nein	Nein	Nein	Nein
Meldeanzeige	Nein	Nein	Nein	Nein
Meldeindikator	Nein	Nein	Nein	Nein
Meldefenster	Nein	Nein	Nein	Nein
Anzeige Kriterienanalyse	Nein	Ja ¹⁾	Ja	Ja
ProDiag-Übersicht	Nein	Ja ¹⁾	Ja	Ja
GRAPH-Übersicht	Nein	Ja ¹⁾	Ja	Ja
PLC-Codeanzeige	Nein	Ja ¹⁾	Ja	Ja
f(x)-Kurvenanzeige	Nein	Nein	Nein	Nein
f(t)-Kurvenanzeige	Nein	Nein	Nein	Nein

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Tabellenanzeige	Nein	Nein	Nein	Nein
Wertetabelle	Nein	Nein	Nein	Nein
Media Player	Nein	Nein	Nein	Nein
Kanaldiagnose	Nein	Nein	Nein	Nein
WLAN-Empfang	Nein	Nein	Nein	Nein
Zonen-Name	Nein	Nein	Nein	Nein
Zonen-Signal	Nein	Nein	Nein	Nein
Wirkbereichs-Name	Nein	Nein	Nein	Nein
Wirkbereichs-Name (RFID)	Nein	Nein	Nein	Nein
Wirkbereichs-Signal	Nein	Nein	Nein	Nein
Ladezustand	Nein	Nein	Nein	Nein
Handrad	Nein	Nein	Ja	Nein
Hilfe-Indikator	Nein	Nein	Nein	Nein
Sm@rtClient-Anzeige	Nein	Nein	Nein	Nein
Status/Steuern	Nein	Nein	Nein	Nein
System-Diagnoseanzeige	Nein	Nein	Nein	Nein
System-Diagnosefenster	Nein	Nein	Nein	Nein

1) Nur Mobile Panels mit einer höheren Geräteversion als 12.0.0.0 unterstützen dieses Bildschirmobjekt

Tabelle 5-3 Dynamisch

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Anzeige	Ja	Ja	Ja	Ja
Bedienbarkeit	Nein	Ja	Ja	Ja
Sichtbarkeit	Ja	Ja	Ja	Ja
Bewegungen	Ja	Ja	Ja	Ja

Tabelle 5-4 Zusätzliche Objekte

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Gruppen	Ja	Ja	Ja	Ja
Softkeys	Ja	Ja	Ja	Ja
Zyklen	Ja	Ja	Ja	Ja
VB-Skripte	Nein	Ja	Ja	Ja
Funktionslisten	Ja	Ja	Ja	Ja
Grafiksammlung	Ja	Ja	Ja	Ja

5.1 TIA Portal Openness-Objekt

Tabelle 5-5 Variablen

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Multiplex-Variablen	Ja	Ja	Ja	Ja
Arrays	Ja	Ja	Ja	Ja
Anwenderdatentypen	Ja	Ja	Ja	Ja
Intern	Nein	Ja	Ja	Ja
Verwendungsstellen von elementaren Datentypen	Ja	Ja	Ja	Ja
Verwendungsstellen von Anwenderdatentypen	Ja	Ja	Ja	Ja
Verwendungsstellen von Arrays	Ja	Ja	Ja	Ja

Zusätzlich unterstützt TIA Portal Openness alle Wertebereiche, die von den Kommunikationstreibern unterstützt werden.

Verbindungen

TIA Portal Openness unterstützt nicht integrierte Verbindungen, die auch von den jeweiligen Bediengeräten unterstützt werden. Weitere Informationen finden Sie in der Online-Hilfe für das TIA Portal unter „Prozesse visualisieren > Mit Steuerungen kommunizieren > Geräteabhängigkeit“.

Tabelle 5-6 Listen

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Textlisten	Ja	Ja	Ja	Ja
Grafiklisten	Ja	Ja	Ja	Ja

Tabelle 5-7 Texte

Objekt	Basic Panels	Comfort Panels	Mobile Panels	RT Advanced
Mehrsprachige Texte	Ja	Ja	Ja	Ja
Formatierte Texte und deren Verwendungsstellen	Nein	Ja	Ja	Ja

5.2 Allgemeine Funktionen

5.2.1 Einleitung

Übersicht

TIA Portal Openness unterstützt eine Auswahl von Funktionen für definierte Aufgaben, die Sie außerhalb des TIA Portals über TIA Portal Openness API aufrufen können.

Hinweis

Wenn bereits eine Vorgängerversion von TIA Portal Openness installiert ist, wird die aktuelle Version parallel installiert.

Die nachstehenden Abschnitte vermitteln Ihnen einen Überblick über die typischen Schritte der Programmierung. Sie erfahren, wie die einzelnen Codeabschnitte interagieren und wie sich die jeweiligen Funktionen zu einem vollständigen Programm verzahnen. Sie erhalten auch einen Überblick über die Codekomponenten, die der jeweiligen Aufgabe angepasst werden müssen.

Beispielprogramm

Die einzelnen Schritte der Programmierung werden mit der Funktion "API-Zugriff in einer Konsolenanwendung" als Beispiel erläutert. Sie als Benutzer integrieren die zur Verfügung gestellten Funktionen in diesen Programmcode und passen die jeweiligen Codekomponenten an diese Aufgabe an.

Funktionen

Im nachstehenden Abschnitt sind die Funktionen für definierte Aufgaben aufgeführt, die Sie mit TIA Portal Openness außerhalb des TIA Portals aufrufen können.

Siehe auch

Einsatzmöglichkeiten (Seite 52)

Objektliste (Seite 69)

5.2.2 Informationen über installierte TIA Portal Openness-Versionen

Voraussetzung

- TIA Portal Openness und TIA Portal sind installiert

Verwendung

Ab TIA Portal Openness V14 hat jede installierte Version einen Registrierungsschlüssel, der Informationen über die Version enthält. Das ermöglicht die automatische Erzeugung von Anwendungskonfigurationsdatei für jede installierte Version von TIA Portal Openness.

Den Registrierungsschlüssel finden Sie unter dem folgendem Pfad:

```
HKEY_LOCAL_MACHINE\Software\Siemens\Automation\Openness\xx.x\PublicAPI
```

Hinweis

Die Versionsnummer in diesem Pfad ist immer die Nummer der gegenwärtig installierten Version des TIA Portals. Bei mehreren parallelen Installationen gibt es mehrere Sätze von Einträgen für TIA Portal Openness in der Registrierung.

Es gibt einen einzigen Schlüssel für jede Version von TIA Portal Openness. Die Namen der Versionen sind die gleichen wie in der beschriebenen Assembly, zum Beispiel die Registrierungseinträge für TIA Portal Openness:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\Openness\xx.x\PublicAPI\xx.x.x.x] "PublicKeyToken"="d29ec89bac048f84"
"Siemens.Engineering"="C:\Program Files\Siemens\Automation\PortalVxx\PublicAPI\Vxx\Siemens.Engineering.dll"
"Siemens.Engineering.Hmi"="C:\Program Files\Siemens\Automation\PortalVxx\PublicAPI\Vxx\Siemens.Engineering.Hmi.dll"
"EngineeringVersion"="xx"
"AssemblyVersion"="xx.x.x.x"
```

Hinweis

Wenn Sie eine Anwendungskonfigurationsdatei (Seite 84) erzeugen möchten, können Sie den Pfad für Siemens.Engineering.dll, Siemens.Engineering.Hmi.dll und Token des öffentlichen Schlüssels dem Registrierungsschlüssel entnehmen.

5.2.3 Unterstützung von IntelliSense durch TIA Portal Openness

Verwendung

Der Intellisense-Support für TIA Portal Openness hilft Ihnen bei verfügbaren Attributen oder Methoden über Tooltip-Informationen. Er kann Aufschluss über Anzahl, Namen und Typen der erforderlichen Parameter geben. Im folgenden Beispiel gibt der fett gedruckte Parameter in der ersten Zeile den nächsten beim Eingeben der Funktion erforderlichen Parameter an.

```
project = tiaPortal.Projects.OpenWithUpgrade(projectInfo);
```

 **Project ProjectComposition.OpenWithUpgrade(FileInfo path)**
Open Action with project update is necessary

Sie können Informationen über Parameter manuell abrufen. Dies geschieht durch Klicken auf "Edit IntelliSense/Parameter Info", durch Betätigen der Tastenkombination CTRL+SHIFT+SPACE oder durch Klicken auf die Schaltfläche "Parameter Info" in der Symbolleiste des Editors.

5.2.4 Standard-Bibliotheken

Fügen Sie die folgenden Namensraumanweisungen am Beginn des jeweiligen Codebeispiels ein, um sicherzustellen, dass die Codebeispiele funktionieren:

```
using System;
using System.Collections.Generic;
using System.IO;
using Siemens.Engineering;
using Siemens.Engineering.Cax;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Compare;
using Siemens.Engineering.Download;
using Siemens.Engineering.Hmi;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.RuntimeScripting;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Extensions;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.HW.Utilities;
using Siemens.Engineering.Library;
using Siemens.Engineering.Library.MasterCopies;
using Siemens.Engineering.Library.Types;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.TechnologicalObjects;
using Siemens.Engineering.SW.TechnologicalObjects.Motion;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Upload;
```


5.2.5 Einsatz der Codebeispiele

Aufbau der Codeausschnitte

Jeder Codeausschnitt in dieser Dokumentation wird als Funktion ohne Rückgabewert mit einem Objektverweis als Übertragungsparameter implementiert. Auf das Beseitigen von Objekten wird zum Zweck der besseren Lesbarkeit verzichtet. Objekte aus dem TIA Portal werden mit ihrem Namen mithilfe der Methode `Find` adressiert.

```
//Deletes a single screen from a user folder or a system folder
private static void DeleteScreenFromFolder(HmiTarget hmiTarget)
{
    //The screen "MyScreen" will be deleted if it is existing in the folder
    "myScreenFolder".
    //If "myScreen" is stored in a subfolder of "myScreenFolder" it will not be deleted.
    string screenName = "MyScreen";
    ScreenUserFolder folder = hmiTarget.ScreenFolder.Folders.Find("myScreenFolder");
    ScreenComposition screens = folder.Screens;
    Screen screen = screens.Find(screenName);
    if (screen != null)
    {
        screen.Delete();
    }
}
```

Zum Ausführen dieses Codeausschnitts benötigen Sie Folgendes:

- Ein WinCC-Projekt mit einem HMI-Gerät, das eine Gruppe mit mindestens einem Bild beinhaltet.
- Eine Funktion, die das Bediengerät instanziiert.

Hinweis

Wenn Sie Verzeichnispfade angeben, verwenden Sie den absoluten Verzeichnispfad, zum Beispiel "C:/path/file.txt".

Relative Verzeichnispfade sind nur in den XML-Dateien für Import und Export zulässig, zum Beispiel "file.txt" oder "C:/path01/.../path02/file.txt".

Beispiel für Ausführung des Codeausschnitts

Verwenden Sie das folgende Beispiel zum Ausführen des Codeausschnitts "DeleteScreenFromFolder" als Teil des Beispielprogramms "Hello TIA":

```
//In the sample program "Hello TIA" replace the function call
//"IterateThroughDevices(project)" by the following functions calls:
    HmiTarget hmiTarget = GetTheFirstHmiTarget(project);
    DeleteScreenFromFolder(hmiTarget);

//Put the following function definitions before or after the
//function definition of "private static void IterateThroughDevices(Project project)":
private static HmiTarget GetTheFirstHmiTarget(Project project)
{
    if (project == null)
    {
        Console.WriteLine("Project cannot be null");
        throw new ArgumentNullException("project");
    }
    foreach (Device device in project.Devices)
    {
        //This example looks for devices located directly in the project.
        //Devices which are stored in a subfolder of the project will not be affected by this
        example.
        {
            foreach (DeviceItem deviceItem in device.DeviceItems)
            {
                DeviceItem deviceItemToGetService = deviceItem as DeviceItem;
                SoftwareContainer container =
                deviceItemToGetService.GetService<SoftwareContainer>();
                if (container != null)
                {
                    HmiTarget hmi = container.Software as HmiTarget;
                    if (hmi != null)
                    {
                        return hmi;
                    }
                }
            }
        }
    }
    return null;
}

//Deletes a single screen from a user folder or a system folder
private static void DeleteScreenFromFolder(HmiTarget hmiTarget)
{
    string screenName = "MyScreen";
    ScreenUserFolder folder = hmiTarget.ScreenFolder.Folders.Find("myScreenFolder");
    ScreenComposition screens = folder.Screens;
    Screen screen = screens.Find(screenName);
    if (screen != null)
    {
        screen.Delete();
    }
}
```

5.2.6 Beispielprogramm

Anwendungsbeispiel: API-Zugriff in einer Anwendung erstellen

Der vollständige Programmcode des Anwendungsbeispiels wird nachstehend aufgeführt. Die typischen Schritte für die Programmierung werden nachfolgend anhand dieses Beispiels beschrieben.

Hinweis

Für das Anwendungsbeispiel ist eine Anwendungskonfigurationsdatei (Seite 84) erforderlich.

5.2 Allgemeine Funktionen

```

using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
namespace HelloTIA
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            RunTiaPortal();
        }

        private static void RunTiaPortal()
        {
            Console.WriteLine("Starting TIA Portal");
            using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
            {
                Console.WriteLine("TIA Portal has started");
                ProjectComposition projects = tiaPortal.Projects;

                Console.WriteLine("Opening Project...");
                // please adapt the path and the extension apx to the installed version of
TIA Portal
                FileInfo projectPath = new FileInfo("C:\\\\Demo\\\\AnyCompanyProject.apx"); //
edit the path according to your project
                Project project = null;
                try
                {
                    project = projects.OpenWithUpgrade(projectPath);
                }
                catch (Exception)
                {
                    Console.WriteLine(String.Format("Could not open project {0}",
projectPath.FullName));
                    Console.WriteLine("Demo complete hit enter to exit");
                    Console.ReadLine();
                }
            }
        }
    }
}

```

```
        return;
    }

    Console.WriteLine(String.Format("Project {0} is open",
project.Path.FullName));

    IterateThroughDevices(project);

    project.Close();

    Console.WriteLine("Demo complete hit enter to exit");
    Console.ReadLine();
}

private static void IterateThroughDevices(Project project)
{
    if (project == null)
    {
        Console.WriteLine("Project cannot be null");
        return;
    }

    Console.WriteLine(String.Format("Iterate through {0} device(s)",
project.Devices.Count));

    foreach (Device device in project.Devices)
    {
        Console.WriteLine(String.Format("Device: \"{0}\".", device.Name));
    }

    Console.WriteLine();
}
}
```

Vorgehensweise in Einzelschritten

1. TIA Portal in der Entwicklungsumgebung bekannt machen

Erzeugen Sie in Ihrer Entwicklungsumgebung einen Verweis auf alle "dll-Dateien" im Verzeichnis "C:\Program Files\Siemens\Automation\PortalV..\PublicAPIV..".

Nachstehend finden Sie eine Beschreibung dieses Vorgangs anhand der Datei "Siemens.Engineering.dll" als Beispiel.

Die Datei "Siemens.Engineering.dll" befindet sich im Verzeichnis "C:\Program Files\Siemens\Automation\PortalV..\PublicAPIV..". Erzeugen Sie in Ihrer Entwicklungsumgebung einen Verweis auf die Datei "Siemens.Engineering.dll".

Hinweis

Weisen Sie dem Parameter "CopyLocal" in den Referenzattributen den Wert "False" zu.

2. Namensraum für das TIA Portal veröffentlichen

Fügen Sie den folgenden Code hinzu:

```
using Siemens.Engineering;
```

3. TIA Portal veröffentlichen und starten

Um das TIA Portal zu veröffentlichen und zu starten, fügen Sie den folgenden Code ein:

```
using (TiaPortal tiaPortal = new TiaPortal())
{
    // Add your code here
}
```

4. Projekt öffnen

Um ein Projekt zu öffnen, können Sie beispielsweise den folgenden Code verwenden:

```
ProjectComposition projects = tiaPortal.Projects;
Console.WriteLine("Opening Project...");
//please adapt the path and the extension apx to the installed version of TIA Portal
FileInfo projectPath = new FileInfo("C:\\Demo\\AnyCompanyProject.apx");
Project project = null;
try
{
    project = projects.Open(projectPath);
}
catch (Exception)
{
    Console.WriteLine(String.Format("Could not open project {0}", projectPath.FullName));
    Console.WriteLine("Demo complete hit enter to exit");
    Console.ReadLine();
    return;
}
Console.WriteLine(String.Format("Project {0} is open", project.Path.FullName));
```

5. Geräte eines Projekts enumerieren

Um alle Geräte des Projekts zu enumerieren, fügen Sie den folgenden Code ein:

```
static private void IterateThroughDevices(Project project)
{
    if (project == null)
    {
        Console.WriteLine("Project cannot be null");
        return;
    }

    Console.WriteLine();
    Console.WriteLine(String.Format("Iterate through {0} device(s)",
project.Devices.Count));
    foreach (Device device in project.Devices)
    {
        Console.WriteLine(String.Format("Device: \"{0}\".", device.Name));
    }
    Console.WriteLine();
}
```

6. Projekt speichern und schließen

Fügen Sie den folgenden Code ein und speichern und schließen Sie das Projekt:

```
project.Save();
project.Close();
```

5.2.7 Programmierschritte

Übersicht

TIA Portal Openness erfordert die folgenden Schritte der Programmierung für den Zugriff mittels TIA Portal Openness API:

1. TIA Portal in der Entwicklungsumgebung bekannt machen
2. Programmzugriff auf das TIA Portal einrichten
3. Programmzugriff auf das TIA Portal aktivieren
4. TIA Portal veröffentlichen und starten
5. Projekt öffnen
6. Befehle ausführen
7. Projekt speichern und schließen
8. Verbindung zum TIA Portal beenden

Hinweis

Zulässige Strings

In Strings im TIA Portal sind nur bestimmte Zeichen zulässig. Alle über die TIA Portal Openness-Anwendung an das TIA Portal übergebene Strings unterliegen diesen Regeln. Wenn Sie ein ungültiges Zeichen in einem Parameter übergeben, wird eine Ausnahme angezeigt.

Siehe auch

Beispielprogramm (Seite 79)

Einsatz der Codebeispiele (Seite 77)

5.2.8 Verbindung zum TIA Portal aufbauen

Einleitung

Sie starten das TIA Portal mit TIA Portal Openness oder stellen die Verbindung zu einem bereits ausgeführten TIA Portal her. Wenn Sie das TIA Portal mit einer TIA Portal Openness-Anwendung starten, geben Sie an, ob das TIA Portal mit oder ohne grafische Benutzeroberfläche gestartet werden soll. Wenn Sie mit dem TIA Portal ohne Benutzeroberfläche arbeiten, wird das TIA Portal nur als Prozess des Betriebssystems gestartet. Sie erstellen mit einer TIA Portal Openness-Anwendung bei Bedarf mehrere Instanzen des TIA Portals.

Hinweis

Wenn Sie über die TIA Portal Openness-Anwendung auf die Oberfläche des TIA Portals zugreifen, können Sie keinen HMI-Editor verwenden. Sie können den Editor „Geräte & Netze“ oder den Programmiereditor manuell oder mit TIA Portal Openness API öffnen.

Zum Starten des TIA Portals mit einer TIA Portal Openness-Anwendung stehen Ihnen folgende Optionen zur Verfügung.

- Verwenden Sie eine Anwendungskonfigurationsdatei (in den meisten Fällen zu empfehlen).
- Verwenden Sie die Methode "AssemblyResolve" (zu empfehlen beim Kopieren usw.).
- Kopieren Sie die Siemens.Engineering.dll in das Verzeichnis der TIA Portal Openness-Anwendung.

Hinweis

Es empfiehlt sich, die Siemens.Engineering.dll mit Hilfe der Anwendungskonfigurationsdatei zu laden. Bei Verwendung dieser Methode werden die starken Namen berücksichtigt und schädliche Änderungen an der Engineering.dll führen zu einem Ladefehler. Bei Verwendung der Methode AssemblyResolve ist dies nicht erkennbar.

Starten des TIA Portals mit einer Anwendungskonfigurationsdatei

Erzeugen Sie in der Anwendungskonfigurationsdatei Verweise auf alle erforderlichen Programmbibliotheken. Die Anwendungskonfigurationsdatei verteilen Sie zusammen mit der TIA Portal Openness-Anwendung.

Speichern Sie die Anwendungskonfigurationsdatei "app.config" im selben Verzeichnis wie die TIA Portal Openness-Anwendung und beziehen Sie diese Datei in Ihre Anwendung ein. Prüfen Sie, ob der Dateipfad in jedem Code mit dem Installationspfad des TIA Portals übereinstimmt.

Für die Anwendungskonfigurationsdatei können Sie den folgenden Codeausschnitt verwenden:

```
<?xml version="1.0"?>
<configuration>
  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <dependentAssembly>
        <assemblyIdentity name="Siemens.Engineering" culture="neutral"
publicKeyToken="d29ec89bac048f84"/>
        <!-- Edit the following path according to your installed version of TIA Portal
-->
        <codeBase version="xx.x.x.x" href="FILE://C:\Program
Files\Siemens\Automation\Portal Vxx\PublicAPI\Vxx\Siemens.Engineering.dll"/>
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="Siemens.Engineering.Hmi" culture="neutral"
publicKeyToken="d29ec89bac048f84"/>
        <!-- Edit the following path according to your installed version of TIA Portal
-->
        <codeBase version="xx.x.x.x" href="FILE://C:\Program
Files\Siemens\Automation\Portal Vxx\PublicAPI\Vxx\Siemens.Engineering.Hmi.dll"/>
      </dependentAssembly>
    </assemblyBinding>
  </runtime>
</configuration>
```

Verwenden Sie den folgenden Programmcode zum Öffnen einer neuen Instanz des TIA Portals über die Anwendungskonfigurationsdatei:

```
//Connect a TIA Portal Openness application via API using
using System;
using System.IO;
using Siemens.Engineering;

namespace UserProgram
{
    internal class MyProgram
    {
        public static void Main(string[] args)
        {
            // To start TIA Portal with user interface:
            // using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
            //
            // To start TIA Portal without user interface:
            // using (TiaPortal tiaPortal = new
TiaPortal(TiaPortalMode.WithoutUserInterface))
            using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
            {
                //begin of code for further implementation
                //...
                //end of code
            }
        }
    }
}
```

Starten des TIA Portals mit der Methode "AssemblyResolve"

Bauen Sie den Programmcode von TIA Portal Openness so auf, dass die Registrierung auf das Ereignis "AssemblyResolve" so früh wie möglich erfolgt. Verkapseln Sie den Zugriff auf das TIA Portal in einem zusätzlichen Objekt oder einer zusätzlichen Methode.

Vorsicht ist beim Auflösen der Engineering Assembly mit einer Assembly-Resolver-Methode geboten. Wenn Typen aus der Engineering Assembly verwendet werden, bevor der Assembly Resolver ausgeführt wurde, stürzt das Programm ab. Der Grund dafür ist, dass der Just-in-time-Übersetzer (JIT-Übersetzer) eine Methode erst dann übersetzt, wenn er sie ausführen muss. Wenn Typen einer Engineering Assembly beispielsweise in "Main" verwendet werden, versucht der JIT-Übersetzer, "Main" bei laufendem Programm zu übersetzen. Das schlägt fehl, weil der JIT-Übersetzer nicht weiß, wo die Engineering Assembly zu finden ist. Die Registrierung des Assembly Resolver in Main ändert daran nichts. Die Methode muss vor der Registrierung des Assembly Resolver laufen und übersetzt werden, bevor der Assembly Resolver ausgeführt werden kann. Die Lösung für dieses Problem besteht darin, die Business Logic, die Typen aus der Engineering Assembly verwendet, in einer separaten Methode unterzubringen. Dabei verwendet die separate Methode nur Typen, die der JIT-Übersetzer bereits versteht. Das vorliegende Beispiel verwendet eine Methode, die "void" zurückgibt, keine Parameter hat und sämtliche Business Logic enthält. Jetzt wird "Main" vom JIT-Übersetzer erfolgreich übersetzt, weil er alle Typen in Main versteht. Wenn RunTiaPortal während der Laufzeit aufgerufen wird, ist der Assembly Resolver bereits registriert. Somit

weiß der JIT-Übersetzer bei dem Versuch, die Typen der Business Logic zu finden, wo sich die Engineering Assembly befindet.

Verwenden Sie den folgenden Programmcode zum Öffnen einer neuen Instanz des TIA Portals.

```
using System;
using System.IO;
using System.Reflection;
using Siemens.Engineering;

namespace UserProgram
{
    static class MyProgram
    {
        public static void Main(string[] args)
        {
            AppDomain.CurrentDomain.AssemblyResolve += MyResolver;
            RunTiaPortal();
        }
        private static void RunTiaPortal()
        {
            // To start TIA Portal with user interface:
            // using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
            //
            // To start TIA Portal without user interface:
            // using (TiaPortal tiaPortal = new
TiaPortal(TiaPortalMode.WithoutUserInterface))
            using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
            {
                //begin of code for further implementation
                //...
                //end of code
            }
        }
        private static Assembly MyResolver(object sender, ResolveEventArgs args)
        {
            int index = args.Name.IndexOf(',');
            if (index == -1)
            {
                return null;
            }
            string name = args.Name.Substring(0, index) + ".dll";
            // Edit the following path according to your installed version of TIA Portal
            string path = Path.Combine(@"C:\Program Files\Siemens\Automation\Portal
Vxx\PublicAPI\Vxx\", name);
            string fullPath = Path.GetFullPath(path);
            if (File.Exists(fullPath))
            {
                return Assembly.LoadFrom(fullPath);
            }
            return null;
        }
    }
}
```

Auf aktive Instanzen des TIA Portals zugreifen

Um die Verbindung zu einer aktiven Instanz des TIA Portals mit einer TIA Portal Openness-Anwendung herstellen zu können, listen Sie zunächst die Instanzen des TIA Portals auf. Innerhalb einer Windows-Sitzung können Sie Verbindungen zu mehreren Instanzen herstellen. Die aktive Instanz kann das TIA Portal mit oder ohne gestartete Benutzeroberfläche sein:

```
foreach (TiaPortalProcess tiaPortalProcess in TiaPortal.GetProcesses())
{
    //...
}
```

Hinweis

Zur Enumeration aller aktiven TIA Portal-Prozesse ist keine Instanz des TIA Portals erforderlich, da es sich bei `GetProcesses` um eine statische Methode für die Klasse `TiaPortal` handelt.

Wenn Sie die Prozess-ID der Instanz des TIA Portals kennen, verwenden Sie die Prozess-ID für den Zugriff auf das Objekt. Der Startvorgang des TIA Portals benötigt eine gewisse Zeit, bevor Sie TIA Portal Openness mit dem TIA Portal verbinden können.

```
Process process = ...;
Thread.Sleep(TimeNeededToStartUp); // The process may not be ready to service requests
immediately.
TiaPortalProcess tiaPortalProcess = TiaPortal.GetProcess(process.Id);
```

Hinweis

Die Prozess-ID, über die der Benutzer verbunden ist, muss ausgeführt werden und einen Zustand erreicht haben, in dem sie Anfragen bezüglich Prozessinformationen beantworten kann.

Beim Herstellen einer Verbindung zu einer aktiven Instanz des TIA Portals erscheint eine Verbindungsaufforderung der TIA Portal Openness-Firewall. Hier können Sie für die Verbindung Folgendes angeben:

- Verbindung einmal erlauben
 - Verbindung nicht erlauben
 - Verbindungen von dieser Applikation immer erlauben
- Weitere Informationen hierzu siehe TIA Portal Openness-Firewall (Seite 90).

Hinweis

Wird die Registry-Eingabeaufforderung dreimal zurückgewiesen, löst das System eine Ausnahme vom Typ `EngineeringSecurityException` aus.

Nach Herstellung der Verbindung zum Prozess können Sie mit Hilfe eines der folgenden Attribute Informationen zu den Instanzen des TIA Portals abrufen:

Attribut	Information
InstalledSoftware as IList<TiaPortalProduct>	Gibt Informationen über die installierten Produkte zurück.
Mode as TiaPortalMode	Gibt den Modus zurück, in dem das TIA Portal gestartet wurde (WithoutUserInterface/ WithUserInterface).
AttachedSessions as IList<TiaPortalSession>	Gibt eine Liste von Anwendungen zurück, die mit dem TIA Portal verbunden sind.
ProjectPath as FileInfo	Gibt den Dateinamen des im TIA Portal geöffneten Projekts einschließlich des Ordners zurück, z. B. "D:\WinCCProjects\ColorMixing\ColorMixing.ap*" Wenn kein Projekt geöffnet ist, wird eine leere Zeichenfolge zurückgegeben.
ID as int	Gibt die Prozess-ID der Instanz des TIA Portals zurück.
Path as FileInfo	Gibt den Pfad zur ausführbaren Datei des TIA Portals zurück.

Hinweis

TIA Portal Openness unterstützt keinen Kompatibilitätsmodus. Jedes im Projekt verwendete Optionspaket oder HSP aus TIA Portal muss für die Openness-Anwendung installiert werden, um geöffnet oder dem Projekt angehängt werden zu können.

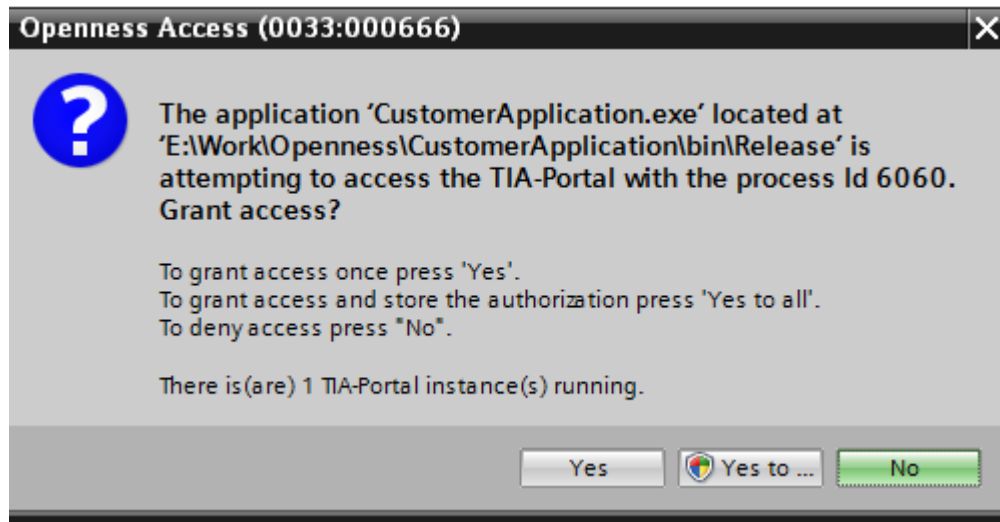
Zum Abruf des aktuellen TIA Portal-Prozesses aus einer TIA Portal-Anwendung durch Aufrufen von GetCurrentProcess() an der Instanz des TIA Portals:

```
TiaPortal tiaPortal = ...;
TiaPortalProcess tiaPortalProcess = tiaPortal.GetCurrentProcess();
```

5.2.9 TIA Portal Openness-Firewall

Eingabeaufforderung der TIA Portal Openness-Firewall

Wenn Sie versuchen, über TIA Portal Openness eine Verbindung zu einem laufenden TIA Portal herzustellen, fordert das TIA Portal Sie auf, die Verbindung zu akzeptieren oder zurückzuweisen, wie im folgenden Screenshot dargestellt.



Verbindung zum TIA Portal einmal erlauben

Wenn Sie Ihre TIA Portal Openness-Anwendung nur einmal mit dem TIA Portal verbinden möchten, klicken Sie bei der Eingabeaufforderung auf "Yes". Wenn Ihre TIA Portal Openness-Anwendung das nächste Mal versucht, eine Verbindung zum TIA Portal herzustellen, wird die Eingabeaufforderung wieder angezeigt.

Whitelist-Eintrag durch Verbinden mit dem TIA Portal hinzufügen

Zum Erzeugen eines Whitelist-Eintrags für Ihre TIA Portal Openness-Anwendung befolgen Sie diese Schritte:

1. Wenn Sie bei der Eingabeaufforderung auf "Yes to all" klicken, wird ein Dialog zur Benutzerkontensteuerung angezeigt.
2. Klicken Sie im Dialog zur Benutzerkontensteuerung auf "Yes". um Ihre Anwendung zur Whitelist in der Windows-Registrierdatenbank hinzuzufügen und um die Anwendung mit dem TIA Portal zu verknüpfen.

Whitelist-Eintrag ohne TIA Portal hinzufügen

Wenn Sie der Whitelist einen Eintrag hinzufügen möchten, ohne dafür das TIA Portal zu verwenden, können Sie eine Registrierungsdatei wie diese erzeugen:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\Openness\xx.x\Whitelist\CustomerApplication.exe]
[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\Openness\xx.x\Whitelist\CustomerApplication.exe\Entry]
"Path"="E:\\Work\\Openness\\CustomerApplication\\bin\\Release\\CustomerApplication.exe"
"DateModified"="2014/06/10 15:09:44.406"
"FileHash"="0rXRKUCNzMWHOMFrT52OwXzqJefl0ran4UykTeBraaY="
```

Das folgende Beispiel zeigt, wie sie den Filehash-Wert und das Datum der letzten Änderung berechnen können:


```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;

using Siemens.Engineering.SW.ExternalSources;

using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security.Cryptography;
namespace TIAPortalOpennessFirewall
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            RunTiaPortal();

            string applicationPath = @"E:\\Work\\Openness\\CustomerApplication\\bin\\Release\\
            CustomerApplication.exe";
            string lastWriteTimeUtcFormatted = String.Empty;
            DateTime lastWriteTimeUtc;
            HashAlgorithm hashAlgorithm = SHA256.Create();
            FileStream stream = File.OpenRead(applicationPath);
            byte[] hash = hashAlgorithm.ComputeHash(stream);
            // this is how the hash should appear in the .reg file
            string convertedHash = Convert.ToBase64String(hash);
            lastWriteTimeUtc = fileInfo.LastWriteTimeUtc; // fileInfo not defined

            // this is how the last write time should be formatted
            lastWriteTimeUtcFormatted = lastWriteTimeUtc.ToString(@"yyyy/MM/dd HH:mm:ss.fff");

        }
    }
    private static void RunTiaPortal()
    {
        Console.WriteLine("Starting TIA Portal");
        using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
        {

```

```
Console.WriteLine("TIA Portal has started");
ProjectComposition projects = tiaPortal.Projects;
Console.WriteLine("Opening Project...");
// please adapt the path and the extension apx to the installed version of TIA Portal
FileInfo projectPath = new FileInfo(@"C:\Demo\AnyCompanyProject.apx");
//edit the path according to your project
Project project = null;
try
{
    project = projects.OpenWithUpgrade(projectPath);
}
catch (Exception)
{
    Console.WriteLine(String.Format("Could not open project {0}", projectPath.FullName));
    Console.WriteLine("Demo complete hit enter to exit");
    Console.ReadLine();
    return;
}
Console.WriteLine(String.Format("Project {0} is open", project.Path.FullName));
IterateThroughDevices(project);
project.Close();
Console.WriteLine("Demo complete hit enter to exit");
Console.ReadLine();
}
}
private static void IterateThroughDevices(Project project)
{
    if (project == null)
    {
        Console.WriteLine("Project cannot be null");
        return;
    }
    Console.WriteLine(String.Format("Iterate through {0} device(s)", project.Devices.Count));
    foreach (Device device in project.Devices)
    {
        Console.WriteLine(String.Format("Device: \"{0}\".", device.Name));
    }
    Console.WriteLine();
}
}
```

5.2.10 Event-Handler

Ereignis-Handler in der TIA Portal Openness-Anwendung

Eine Instanz des TIA Portals bietet die folgenden Ereignisse, auf die Sie mit einem Ereignis-Handler in einer TIA Portal Openness-Anwendung reagieren können. Sie können auf die Attribute von Benachrichtigungen zugreifen und die Antworten entsprechend definieren.

Ereignis	Antwort
Disposed	Mit diesem Ereignis reagieren Sie auf das Schließen des TIA Portals mit einer TIA Portal Openness-Anwendung.
Notification	Mit diesem Ereignis reagieren Sie auf Benachrichtigungen des TIA Portals mit einer TIA Portal Openness-Anwendung. Benachrichtigungen müssen lediglich quittiert werden, zum Beispiel mit "OK".
Confirmation	Mit diesem Ereignis reagieren Sie auf Bestätigungen des TIA Portals mit einer TIA Portal Openness-Anwendung. Bestätigungen erfordern immer eine Entscheidung, zum Beispiel "Möchten Sie das Projekt speichern?".

Programmcode

Ändern Sie den folgenden Programmcode, um Ereignis-Handler in einer TIA Portal Openness-Anwendung zu registrieren:

```
//Register event handler for Disposed-Event
....
    tiaPortal.Disposed +=TiaPortal_Disposed;
....

private static void TiaPortal_Disposed(object sender, EventArgs e)
{
    ....
}

//Register event handler for Notification-Event
....
    tiaPortal.Notification += TiaPortal_Notification;
....

private static void TiaPortal_Notification(object sender, NotificationEventArgs e)
{
    ....
}

//Register event handler for Confirmation-Event
....
    tiaPortal.Confirmation += TiaPortal_Confirmation;
....

private static void TiaPortal_Confirmation(object sender, ConfirmationEventArgs e)
{
    ....
}
```

Attribute von Benachrichtigungen des TIA Portals

Benachrichtigungen des TIA Portals haben die folgenden Attribute:

Attribut	Beschreibung
Caption	Gibt den Namen der Bestätigung zurück.
DetailText	Gibt den Detailtext der Bestätigung zurück.
Icon	Gibt das Symbol der Bestätigung zurück.
IsHandled	Gibt die Bestätigung zurück oder gibt an, ob die Bestätigung noch aussteht.
Text	Gibt den Text der Bestätigung zurück.

Attribute von Bestätigungen

Bestätigungen haben die folgenden Attribute:

Attribut	Beschreibung
Caption	Gibt den Namen der Bestätigung zurück.
Choices	Gibt die Option zum Quittieren der Bestätigung zurück.
DetailText	Gibt den Detailtext der Bestätigung zurück.
Icon	Gibt das Symbol der Bestätigung zurück.
IsHandled	Gibt die Bestätigung zurück oder gibt an, ob die Bestätigung noch aussteht.
Result	Gibt das Ergebnis der Quittierung zurück oder gibt es an.
Text	Gibt den Text der Bestätigung zurück.

Siehe auch

Dialoge mit Systemmeldungen programmgesteuert bestätigen (Seite 97)

5.2.11 Dialoge mit Systemmeldungen programmgesteuert bestätigen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Ereignis-Handler sind registriert.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)

Verwendung

Wenn Sie das TIA Portal mit der Benutzeroberfläche bedienen, werden für einige Programmabläufe Dialoge mit Systemereignissen angezeigt. Anhand dieser Systemereignisse entscheiden Sie, wie Sie fortfahren möchten.

Wenn mit einer TIA Portal Openness-Anwendung auf das TIA Portal zugegriffen wird, müssen diese Systemereignisse über entsprechende „.NET“-Ereignisse quittiert werden.

Die zulässigen Bestätigungen sind in der Liste `Choices` enthalten:

- `Abort`
- `Cancel`
- `Ignore`
- `No`
- `NoToAll`
- `None`

5.2 Allgemeine Funktionen

- OK
- Retry
- Yes
- YesToAll

Der Wert von `ConfirmationEventArgs.Result` muss einer der oben aufgeführten Einträge sein. Andernfalls wird eine Ausnahme ausgelöst.

Programmcode

Ändern Sie den folgenden Programmcode, um den Projekteur über ausgeführte Aktionen einer TIA Portal Openness-Anwendung zu benachrichtigen:

5.2 Allgemeine Funktionen

```

using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;

using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
Namespace
ConfirmDialogsWithSystemMessagesProgrammatically
{
class internal program()
{
private static void Main(string[] args)
{
// To respond to a confirmation event
using (TiaPortal tiaPortal = new TiaPortal());
{
tiaPortal.Confirmation += TiaPortalConfirmation(object sender, ConfirmationEventArgs e);
tiaPortal.Notification += Notification;
try
{
//perform actions that will result in a notification event
}
finally
{
tiaPortal.Notification -= Notification;
}
}
private void TiaPortalConfirmation(object sender, ConfirmationEventArgs e)
{
...
}
//Handles notifications
using (TiaPortal tiaPortal = new TiaPortal())
{
tiaPortal.Notification += Notification;
try
{
//perform actions that will result in a notification event

```



```
}  
finally  
{  
tiaPortal.Notification -= Notification;  
}  
}  
}  
}
```

5.2.12 Verbindung zum TIA Portal beenden

Einleitung

Wenn Sie die TIA Portal-Instanz ohne eine Benutzeroberfläche gestartet haben und Ihre Anwendung der einzige mit dem TIA Portal verknüpfte TIA Portal Openness-Client ist, können Sie die Instanz des TIA Portals mit der TIA Portal Openness-Anwendung schließen. Andernfalls trennen Sie die TIA Portal Openness-Anwendung von der TIA Portal-Instanz.

Trennen oder schließen Sie die aktive Instanz des TIA Portals mit der Methode `IDisposable.Dispose()`.

Die Methode `IDisposable.Dispose()` können Sie wie folgt verwenden:

- Mit einem using-Statement.
- Umgeben Sie die Objektbeschreibung mit einem try-finally-Block und rufen Sie die Methode `IDisposable.Dispose()` innerhalb des finally-Blocks auf.

Wenn Sie die aktive Instanz des TIA Portals beenden, können Sie nicht mehr auf das TIA Portal zugreifen.

Hinweis

Wenn ein Projekteur die TIA Portal-Instanz trotz laufenden Zugriffs einer TIA Portal Openness-Anwendung schließt, wird beim nächsten Zugriff in der TIA Portal Openness-Anwendung eine Ausnahme der Klasse "NonRecoverableException" ausgelöst. Sie können ein Disposed-Ereignis abonnieren, um einen Aufruf beim Schließen des TIA Portals zu erhalten.

Programmcode

Ändern Sie den folgenden Programmcode, um die Verbindung zum TIA Portal zu trennen oder zu schließen:

```
// Add code to dispose the application if the application is still instantiated  
if (tiaPortal != null)  
{  
    tiaPortal.Dispose();  
}
```

Siehe auch

Event-Handler (Seite 95)

5.2.13 Diagnoseschnittstellen im TIA Portal**Verwendung**

Sie können bestimmte Diagnoseinformationen von laufenden Instanzen des TIA Portals über eine statische Methode abrufen. Die Diagnoseschnittstelle ist im Objekt `TiaPortalProcess` implementiert und dieses Objekt kann für jede aktuell laufende Instanz des TIA Portals abgerufen werden.

Die Diagnoseschnittstelle ist nicht gesperrt. Sie können also auf das Objekt `TiaPortalProcess` abrufen und auf seine Mitglieder zugreifen, und zwar unabhängig davon, ob das TIA Portal belegt ist oder nicht. Die Diagnoseschnittstelle umfasst folgende Mitglieder:

Klasse `TiaPortalProcess`

Mitglied	Typ	Funktion
<code>AcquisitionTime</code>	<code>DateTime</code>	Zeitpunkt, zu dem das <code>TiaPortalProcess</code> -Objekt erfasst wurde. Da das Objekt <code>TiaPortalProcess</code> eine vollkommen statische Momentaufnahme vom Zustand des TIA Portals zu einem gegebenen Zeitpunkt darstellt, können darin enthaltene Informationen veraltet sein.
<code>Attach</code>	<code>TiaPortal</code>	Hängt sich dem gegebenen <code>TiaPortalProcess</code> an und gibt eine Instanz des TIA Portals zurück.
<code>AttachedSessions</code>	<code>ICollection<TiaPortalSession></code>	Eine Sammlung aller anderen Sitzungen, die gegenwärtig an dasselbe TIA Portal angehängt sind. Diese Sammlung kann leer sein. Jede Sitzung wird durch ein nachstehend beschriebenes <code>TiaPortalSession</code> -Objekt repräsentiert.

Mitglied	Typ	Funktion
Attaching	EventHandler<AttachingEventArgs>	Dieses Ereignis ermöglicht einer Anwendung, Versuche zum Anhängen an das TIA Portal zuzulassen. Wenn eine andere Anwendung versucht, sich dem TIA Portal anzuhängen, werden die Teilnehmer dieses Ereignisses benachrichtigt. Die Teilnehmer haben dann 10 Sekunden Zeit, um dieser Anbindung zuzustimmen. Wenn ein Teilnehmer dieses Ereignis ignoriert oder nicht rechtzeitig reagiert, gilt das als Ablehnung und der anderen Anwendung wird das Anhängen nicht erlaubt. Abgestürzte Anwendungen, die nicht in der Lage sind, auf dieses Ereignis zu reagieren, können einer Anwendung die Erlaubnis zum Anhängen nicht verweigern.
Dispose	void	Schließt die betreffende Instanz des TIA Portals.
Id	int	Die Prozess-ID des TIA Portals.
InstalledSoftware	ICollection<TiaPortalProduct>	Eine Sammlung aller gegenwärtig installierten Produkte als Teil des TIA Portals. Jedes Produkt wird durch ein nachstehend beschriebenes TiaPortalProduct-Objekt dargestellt.
Mode	TiaPortalMode	Gibt den Modus zurück, in dem das TIA Portal gestartet wurde. Die aktuellen Werte sind WithUserInterface und WithoutUserInterface.
Path	FileInfo	Der Pfad zur ausführbaren Datei des TIA Portals.
ProjectPath	FileInfo	Der Pfad zu dem Projekt, das gegenwärtig im TIA Portal geöffnet ist. Wenn kein Projekt geöffnet ist, hat dieses Attribut den Wert Null.

Klasse TiaPortalSession

Mitglied	Typ	Funktion
AccessLevel	TiaPortalAccessLevel	Zugriffsebene der Sitzung. Wird dargestellt als Merker-Enumeration, wobei mehrere Zugriffsebenen möglich sind. TiaPortalAccessLevel wird nachstehend ausführlich beschrieben.
AttachTime	DateTime	Der Zeitpunkt, zu dem die Verbindung mit dem TIA Portal hergestellt wurde.
Id	int	Die ID der aktuellen Sitzung.
IsActive	bool	Gibt "wahr" zurück, wenn das TIA Portal gegenwärtig einen Aufruf aus dieser Sitzung verarbeitet.
Dispose	void	Trennt die Verbindung des Prozesses zum TIA Portal. Diese Methode erzwingt nicht die Beendigung des Prozesses selbst, so wie es mit System.Diagnostics.Process.Kill der Fall wäre. Die Anwendung, deren Verbindung beendet wird, erhält dennoch ein Disposed-Ereignis. Es erfolgt aber kein weiterer Hinweis darauf, weshalb die Verbindung beendet wurde.
ProcessId	int	Die Prozess-ID des angehängten Prozesses.
ProcessPath	FileInfo	Der Pfad der ausführbaren Datei des angehängten Prozesses.
TrustAuthority	TiaPortalTrustAuthority	Gibt an, ob die aktuelle Sitzung von einem signierten Prozess gestartet wurde und ob es ein TIA Portal Openness-Zertifikat ist oder nicht. TrustAuthority ist eine Merker-Enumeration und wird nachstehend beschrieben.
UtilizationTime	TimeSpan	Zeitraum, den der Prozess aktiv im TIA Portal verbracht hat. Kombiniert mit dem Attribut AttachTime kann dies dazu genutzt werden, prozentuale Nutzungsdaurenteile oder ähnliche Daten zu ermitteln.
Version	string	Die Version der Siemens.Engineering.dll, der die Sitzung angehängt wird.

Enum TiaPortalAccessLevel

Enumerationswert	Funktion
None	Dies ist kein gültiger Wert. Der Wert ist aufgeführt, weil TiaPortalAccessLevel eine Merkerenumeration ist und als solche einen entsprechenden "Nullwert" benötigt, um darzustellen, dass keine Merker gesetzt sind. Aber der Wert erscheint nie im tatsächlichen Gebrauch, weil keine Sitzung ohne Zugriff gestartet werden kann.
Published	Die Sitzung hat Zugriff auf publizierte Funktionalität.
Modify	Die Sitzung hat Änderungszugriff.

Enum TiaPortalTrustAuthority

Enumerationswert	Funktion
None	Das Hauptmodul des angehängten Prozesses ist nicht mit einem Zertifikat signiert.
Signed	Das Hauptmodul ist mit einem Zertifikat signiert, es ist aber kein TIA Portal Openness-Zertifikat.
Certified	Das Hauptmodul ist mit einem TIA Portal Openness-Zertifikat signiert.
CertifiedWithExpiration	Das Hauptmodul ist mit einem TIA Portal Openness-Zertifikat signiert, das seine Gültigkeit mit Ablauf seiner Lebensdauer verliert.

Klasse TiaPortalProduct

Mitglied	Typ	Funktion
Name	string	Der Name des Produkts (z. B. STEP 7 Professional).
Options	IList<TiaPortalProduct>	Eine Sammlung aller zum verbundenen TIA Portal gehörenden Optionspakete, dargestellt als TiaPortalProduct-Objekte. Wenn ein Optionspaket selbst Optionspakete hat, kann sich diese Schachtelung fortsetzen.
Version	string	Der Versionsstring des Produkts.

Der folgende Codeausschnitt bietet ein Beispiel dafür, wie die Diagnoseschnittstelle zum Abfragen von Informationen verwendet wird und wie diese Informationen anschließend in Ihrer Applikation genutzt werden.

```
public void TiaPortalDiagnostics()
{
    IList<TiaPortalProcess> tiaPortalProcesses = TiaPortal.GetProcesses();
    foreach (TiaPortalProcess tiaPortalProcess in tiaPortalProcesses)
    {
        Console.WriteLine("Process ID: {0}", tiaPortalProcess.Id);
        Console.WriteLine("Path: {0}", tiaPortalProcess.Path);
        Console.WriteLine("Project: {0}", tiaPortalProcess.ProjectPath);
        Console.WriteLine("Timestamp: {0}", tiaPortalProcess.AcquisitionTime);
        Console.WriteLine("UI Mode: {0}", tiaPortalProcess.Mode);
        //See method body below.
        Console.WriteLine("Installed Software:");
        EnumerateInstalledProducts(tiaPortalProcess.InstalledSoftware);
        Console.WriteLine("Attached Openness Applications:");
        foreach (TiaPortalSession session in tiaPortalProcess.AttachedSessions)
        {
            Console.WriteLine("Process: {0}", session.ProcessPath);
            Console.WriteLine("Process ID: {0}", session.ProcessId);
            DateTime attachTime = session.AttachTime;
            TimeSpan timeSpentAttached = DateTime.Now - attachTime;
            TimeSpan utilizationTime = session.UtilizationTime;
            long percentageTimeUsed = (utilizationTime.Ticks / timeSpentAttached.Ticks) *
100;

            Console.WriteLine("AttachTime: {0}", attachTime);
            Console.WriteLine("Utilization Time: {0}", utilizationTime);
            Console.WriteLine("Time spent attached: {0}", timeSpentAttached);
            Console.WriteLine("Percentage of attached time spent using TIA Portal: {0}",
percentageTimeUsed);
            Console.WriteLine("AccessLevel: {0}", session.AccessLevel);
            Console.WriteLine("TrustAuthority: {0}", session.TrustAuthority);
            if ((session.TrustAuthority & TiaPortalTrustAuthority.Certified) !=
TiaPortalTrustAuthority.Certified)
            {
                Console.WriteLine("TrustAuthority doesn't match required level, attempting
to terminate connection to TIA Portal.");
                session.Dispose();
            }
        }
    }
}

public void EnumerateInstalledProducts(IEnumerable<TiaPortalProduct> products)
{
    foreach (TiaPortalProduct product in products)
    {
        Console.WriteLine("Name: {0}", product.Name);
        Console.WriteLine("Version: {0}", product.Version);
        //recursively enumerate all option packages
        Console.WriteLine("Option Packages \n:");
        EnumerateInstalledProducts(product.Options);
    }
}
```

Sicherheitsrelevante Information

Aufgrund der Tatsache, dass für die Nutzung der Diagnoseschnittstelle keine Verbindung zum TIA Portal gebraucht wird, ist es möglich, einen Windows-Dienst zu schreiben, der das sich anhängende Ereignis dazu nutzt, jede Anwendung, die sich an ein TIA Portal anzuhängen versucht, zu prüfen. So kann zum Beispiel festgelegt werden, dass nur Anwendungen, die mit dem Namen Ihres Unternehmens beginnen, sich anhängen dürfen. Eine andere Option könnte sein, immer Zugriff zu gewähren, aber Informationen über sich anhängende Prozesse in ein Protokoll zu schreiben. Der folgende Programmcode ist ein Beispiel für einen Ereignis-Handler zum Prüfen eingehender Verbindungen:

```
public void OnAttaching(object sender, AttachingEventArgs e)
{
    string name = Path.GetFileNameWithoutExtension(e.ProcessPath);
    TiaPortalAccessLevel requestedAccessLevel = e.AccessLevel &
TiaPortalAccessLevel.Published;
    TiaPortalTrustAuthority certificateStatus = e.TrustAuthority
&TiaPortalTrustAuthority.Certified;
    if (requestedAccessLevel == TiaPortalAccessLevel.Published &&
        certificateStatus == TiaPortalTrustAuthority.Certified &&
        name.StartsWith("SampleCustomerName"))
    {
        e.GrantAccess();
    }
}
```

5.2.14 Ausschließlicher Zugriff

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Zum Einrichten eines exklusiven Prozesses auf einen angehängten Prozess des TIA Portals bietet die Klasse "TIA Portal" die Methode "ExclusiveAccess(String text)". Die Nutzung eines exklusiven Zugriffs wird dringend empfohlen, auch wenn er nicht obligatorisch ist.

Verwenden Sie "ExclusiveAccess" in einer "using"-Anweisung, um sicherzustellen, dass der Zugriff ordnungsgemäß beendet wird, wenn Ausnahmen auftreten oder die Anwendung heruntergefahren wird.

Hinweis

Jeder Versuch, einen zweiten exklusiven Zugriff innerhalb des Geltungsbereichs eines geöffneten exklusiven Zugriffs zu erzeugen, resultiert in der Auslösung einer wiederherstellbaren Ausnahme.

Ändern Sie das folgende Beispiel, um "ExclusiveAccess" auf eine Instanz zu erhalten:

```
...
[assembly: AssemblyTitle("MyApplication")]
// This will be used for the exclusive access dialog when present....
TiaPortal tiaPortal = ...;
using (ExclusiveAccess exclusiveAccess = tiaPortal.ExclusiveAccess("My Activity"))
{
    ...
}
```

Nach dem Erfassen einer "ExclusiveAccess"-Instanz für einen gegebenen Prozess des TIA Portals wird ein Dialog angezeigt. Dieser Dialog zeigt die bei der Instanziierung vorgesehene Meldung an. Darüber hinaus werden die folgenden Informationen über die Client-Anwendung angezeigt:

- der Assembly-Titel aus den Manifestdaten, wenn verfügbar, andernfalls der Prozessname
- die Prozess-ID
- die SID (Openness-Session-ID der TIA-Portal-Instanz der Client-Anwendung)

Hinweis

Für eine gegebene TIA Portal Openness-Client-Anwendung können mehrere Sitzungen aktiv sein, weil es mehrere Instanzen des TIA Portals gibt, wobei diese jeweils demselben TIA Portal-Prozess zugeordnet sind.

Die Client-Anwendung kann den angezeigten Inhalt des Dialogs für exklusiven Zugriffs auch aktualisieren, indem das Attribut "Text" mit neuen Werten besetzt wird. Ändern Sie den folgenden Programmcode, um dieses Verhalten herbeizuführen:

```
exclusiveAccess = ...;
...
exclusiveAccess.Text = "My Activity Phase 1";
...
exclusiveAccess.Text = "My Activity Phase 2";
...
exclusiveAccess.Text = String.Empty; // or null;
...
```


Sie können verlangen, dass der exklusive Zugriff mit der Schaltfläche „Cancel“/„Abbrechen“ aufgehoben wird. Ändern Sie den folgenden Programmcode, um dieses Verhalten herbeizuführen:

```
exclusiveAccess = ...;
...
if (exclusiveAccess.IsCancellationRequested)
{
    // stop your activity
    ...
}
else
{
    // continue your activity
    ...
}
...
```

5.2.15 Dynamisches Verhalten

Dynamisches Verhalten in Openness

Die Openness-Benutzer können Objekte und ihre Attribute während der Laufzeit festlegen. Sie können außerdem Aktionen aufrufen und bidirektional in der Hierarchie zu zugehörigen Objekten navigieren. Um dynamisches Verhalten zu unterstützen, werden von Openness-Objekten die folgenden Schnittstellen bereitgestellt und implementiert:

IEngineeringInstance

Die Schnittstelle, die im Objektmodell Aufwärtsnavigation ermöglicht.

Eigenschaften:

- IEngineeringObject Parent – übergeordnetes Objekt einer Instanz.

IEngineeringCompositionOrObject

Die Schnittstelle wird von Engineering-Objekten und Zusammenstellungen implementiert.

IEngineeringObject

Die Schnittstelle wird von der Mehrheit der Objekte implementiert, die für die Openness-Benutzer verfügbar sind. Die Schnittstelle IEngineeringObject wird von IEngineeringCompositionOrObject und IEngineeringInstance abgeleitet:

Methoden:

- `IList<EngineeringCreationInfo> GetCreationInfos(string compositionName)` – ruft die Liste der für das Objekt verfügbaren Zusammenstellungsinfos ab.
- `IEngineeringObject Create(string compositionName, Type type, IEnumerable<KeyValuePair<string, object>> parameters)` – erstellt ein `IEngineeringObject` des angegebenen Typs, das mit Werten wie dem angegebenen `compositionName` innerhalb der Parameter initialisiert wird.
- `object GetAttribute(string name)` – ruft ein Attribut mit dem angegebenen Namen ab.
- `IList<EngineeringAttributeInfo> GetAttributeInfos()` – gibt eine Sammlung von `EngineeringAttributeInfo`-Objekten zurück, die die verschiedenen Attribute zu diesem Objekt beschreiben.
- `IList<object> GetAttributes(IEnumerable<string> names)` – ruft eine Liste von Attributen für die angegebenen Namen ab.
- `IEngineeringCompositionOrObject GetComposition(string name)` – ruft ein `IEngineeringCompositionOrObject` mit dem angegebenen Namen ab.
- `IList<EngineeringInvocationInfo> GetInvocationInfos()` – gibt eine Sammlung von `EngineeringAttributeInfo`-Objekten zurück, die die verschiedenen Aktionen zu diesem Objekt beschreiben.
- `object Invoke(string name, IEnumerable<KeyValuePair<Type, object>> parameters)` – ruft die von der aktuellen Instanz repräsentierte Methode unter Verwendung der angegebenen Parameter auf.
- `void SetAttribute(string name, object value)` – legt ein Attribut mit dem angegebenen Namen und dem angegebenen Wert fest.
- `void SetAttributes(IEnumerable<KeyValuePair<string, object>> attributes)` – legt die Attribute mit den angegebenen Namen und den angegebenen Werten wie in den Attributen angegeben fest.
- `EngineeringObjectHandle GetHandle()` – ruft den eindeutigen Handle des Objekts ab.
- `void SetAttributes(IEnumerable<KeyValuePair<string, object>> attributes, AttributeDelegate errorHandler)` – legt die Attribute mit den angegebenen Namen und den angegebenen Werten wie im Schlüsselwertepaar des Attributs angegeben fest. Tritt ein Fehler auf (z. B. ungültiger Attributname/-wert), wird ein Rückruf ausgelöst und Sie können entscheiden, ob sie diesen ignorieren und die Werteinstellung für das nächste Attribut fortsetzen oder die Einstellung der Werte für die restlichen Attribute abbrechen wollen.

Hinweis

In TIA Portal Openness V18 wird eine neue Methode `void SetAttributes(IEnumerable<KeyValuePair<string, object>> attributes, AttributeDelegate errorHandler)` in `IEngineeringObject` hinzugefügt.

Aktuell wird `SetAttributes(IEnumerable<KeyValuePair<string, object>> attributes, AttributeDelegate errorHandler)` von den folgenden Engineering-Objekten unterstützt:

- `Siemens.Engineering.HW.Features.NetworkInterface`
- `Siemens.Engineering.HW.Features.NetworkPort`

- Siemens.Engineering.HW.Features.PlcAccessLevelProvider
- Siemens.Engineering.HW.Address
- Siemens.Engineering.HW.Channel
- Siemens.Engineering.HW.Device
- Siemens.Engineering.HW.DeviceImpl
- Siemens.Engineering.HW.DeviceItem
- Siemens.Engineering.HW.DeviceItemImpl
- Siemens.Engineering.HW.HardwareObject
- Siemens.Engineering.HW.IoConnector
- Siemens.Engineering.HW.IoController
- Siemens.Engineering.HW.IoSystem
- Siemens.Engineering.HW.MrpDomain
- Siemens.Engineering.HW.Node
- Siemens.Engineering.HW.Subnet
- Siemens.Engineering.HW.SyncDomain
- Siemens.Engineering.HW.TransferArea

Wird ein anderes Engineering-Objekt verwendet, verhält sich die Methode wie die Methode `void SetAttributes(IEnumerable<KeyValuePair<string, object>> attributes)`, die keinen Rückruf hat.

Programmcode: Methode SetAttributes(IEnumerable<KeyValuePair<string, object>>)

```

...
...
List<KeyValuePair<string, object>> attributeKeyValuePair = new List<KeyValuePair<string,
object>>()
{
new KeyValuePair<string, object>("Name", "DeviceItemName")
};
deviceItem.SetAttributes(attributeKeyValuePair, (AttributeConfiguration
attributeConfiguration) =>
{
//This callback delegate will be invoked when there is an error while executing the
SetAttributes API,
//reason could be either due to the attribute having invalid attribute name, invalid value,
//invalid type of the value or when tried to set the value to a readonly attribute
switch (attributeConfiguration)
{
case AttributeReadOnly attributeReadOnly:
attributeReadOnly.CurrentSelection = AttributeChoiceSelection.Ignore;
break;
case AttributeNameUnsupported attributeNameUnsupported:
//Attribute name is not supported
attributeNameUnsupported.CurrentSelection = AttributeChoiceSelection.Ignore;
// Or AttributeChoiceSelection.Abort
break;
case AttributeTypeUnsupported attributeTypeUnsupported:
//The the type of supplied value is not supported
attributeTypeUnsupported.CurrentSelection = AttributeChoiceSelection.Ignore; // Or
AttributeChoiceSelection.Abort
break;
case AttributeValueUnsupported attributeValueUnsupported:
//The value is not supported
attributeValueUnsupported.CurrentSelection = AttributeChoiceSelection.Ignore; // Or
AttributeChoiceSelection.Abort
break;
}
}
);

```

IEngineeringRoot

Diese Schnittstelle wird von TiaPortal-Objekten implementiert und gibt an, dass der Benutzer am obersten Objekt in der Hierarchie ist. IEngineeringRoot wird von IEngineeringObject, IEngineeringCompositionOrObject und IEngineeringInstance abgeleitet.

Methoden:

- IEngineeringObject GetObject(EngineeringObjectHandle objectHandle) – ruft das Objekt aus einem Handle ab.

IEngineeringComposition

Die Schnittstelle wird von der Mehrheit der Objekte implementiert, die für die Openness-Benutzer verfügbar sind. Die Schnittstelle IEngineeringObject wird von IEngineeringCompositionOrObject und IEngineeringInstance abgeleitet: Neben den unter Mit Zusammensetzungen arbeiten (Seite 115) beschriebenen Eigenschaften und Methoden werden die folgenden Methoden unterstützt.

Methoden:

- IEngineeringObject Create(Type type, IEnumerable<KeyValuePair<string, object>> parameters) – erstellt ein IEngineeringObject des angegebenen Typs, das mit Werten wie in den Parametern angegeben initialisiert wird.
- IList<EngineeringCreationInfo> GetCreationInfos() – ruft die Sammlung von EngineeringCreateInfo-Objekten ab, die die verschiedenen CreateInfos zu diesem Objekt beschreiben.
- IList<EngineeringInvocationInfo> GetInvocationInfos() – gibt eine Sammlung von EngineeringAttributInfo-Objekten zurück, die die verschiedenen Aktionen zu diesem Objekt beschreiben.
- object Invoke(string name, IEnumerable<KeyValuePair<Type, object>> parameters) – ruft die von der aktuellen Instanz repräsentierte Methode unter Verwendung der angegebenen Parameter auf.

IEngineeringObjectAssociation

Diese Schnittstelle wird von der Sammlung zugehöriger Engineering-Objekte implementiert. Diese Schnittstelle wird von IEngineeringAssociation und IEngineeringInstance abgeleitet. Siehe Arbeiten mit Assoziationen (Seite 114)

IEngineeringServiceProvider

Die Schnittstelle wird von Engineering-Objekten implementiert, die Dienste bereitstellen können.

Methoden:

- T GetService<T>() where T : class, IEngineeringService – ruft eine Instanz vom Typ T ab.
- IList<EngineeringServiceInfo> GetServiceInfos() – gibt eine Sammlung von EngineeringServiceInfo-Objekten zurück, die die verschiedenen Dienste zu diesem Objekt beschreiben.

IEngineeringService

Die Schnittstelle stellt einen Engineering-Dienst dar, der von IEngineeringServiceProvider bereitgestellt wird.

EngineeringObjectHandle

Die Struktur, die einen eindeutigen Objekt-Handle liefert.

Siehe auch

Mit Zusammensetzungen arbeiten (Seite 115)

Arbeiten mit Assoziationen (Seite 114)

5.2.16 Arbeiten mit Assoziationen

Auf Zuordnungen zugreifen

Eine Zuordnung beschreibt die Beziehung zwischen zwei oder mehr Objekten auf Typebene.

TIA Portal Openness unterstützt den Zugriff auf Zuordnungen über den Index und "foreach"-Schleifen. Direktzugriff, beispielsweise über string name, wird nicht unterstützt.

Attribute

Die folgenden Attribute sind verfügbar:

- `int Count`
- `bool IsReadOnly`
- `IEngineeringObject Parent`
- `retType this [ int index ] { get; }`

Methoden

TIA Portal Openness unterstützt die folgenden Methoden:

- `int IndexOf ( type )`: Gibt den Index in der Zuordnung für eine übertragene Instanz zurück.
- `bool Contains ( type )`: Ermittelt, ob die übertragene Instanz in der Zuordnung enthalten ist.
- `IEnumerator GetEnumerator <retType>()`: Kommt in foreach-Schleifen zum Einsatz und ermöglicht den Zugriff auf ein Objekt.
- `void Add ( type )1`: Fügt die übertragene Instanz der Zuordnung hinzu.
- `void Remove ( type )1`: Entfernt die übertragene Instanz aus der Zuordnung.

¹: Wird nicht von allen Zuordnungen unterstützt.

5.2.17 Mit Zusammensetzungen arbeiten

Zusammensetzungen aufrufen

Eine Zusammensetzung ist der Sonderfall einer Zuordnung. Eine Zusammensetzung drückt eine semantische Beziehung zwischen zwei Objekten aus, wobei eines der Objekte ein Teil des anderen ist.

Attribute

Die folgenden Attribute sind verfügbar:

- `int Count`
- `bool IsReadOnly`
- `IEngineeringObject Parent`
- `retType this [int index] {get;}`: Indizierter Zugriff auf ein Objekt der Zusammensetzung.
Diese Art von Zugriff darf nur gezielt genutzt werden, da jede indizierte Zugriffsoperation über Prozessgrenzen hinausgeht.

Methoden

TIA Portal Openness unterstützt die folgenden Methoden:

- `retType Create (id, ...)`: Erzeugt eine neue Instanz und fügt diese Instanz der Zusammensetzung hinzu.
Die Signatur der Methode ist abhängig von der Art und Weise, in der die Instanz erzeugt wird. Diese Methode wird nicht von allen Zusammensetzungen unterstützt.
- `type Find (id, ...)`: Durchsucht eine Zusammensetzung nach der Instanz mit der übertragenen ID.
Die Suche ist nicht rekursiv. Die Signatur der Methode ist abhängig von der Weise, in der nach der Instanz gesucht wird. Diese Methode wird nicht von allen Zusammensetzungen unterstützt.
- `IEnumerator GetEnumerator<retType> ()`: Kommt in foreach-Schleifen zum Einsatz und ermöglicht den Zugriff auf ein Objekt.
- `Delete (type)1`: Löscht die von der aktuellen Objektreferenz spezifizierte Instanz.
- `int IndexOf (type)`: Gibt den Index in der Zusammensetzung für eine übertragene Instanz zurück.
- `bool Contains (type)`: Ermittelt, ob die übertragene Instanz in der Zusammensetzung enthalten ist.
- `void Import(string path, ImportOptions importOptions)1`: Wird für jede Zusammensetzung verwendet, die importierbare Typen enthält.
Jede Importsignatur enthält einen Konfigurationsparameter des Typs "ImportOptions (Seite 1255)" ("None", "Overwrite"), mit dem der Benutzer das Importverhalten steuert.

¹: Nicht von allen Zusammensetzungen unterstützt.

5.2.18 Objektgleichheit prüfen

Verwendung

Als Anwender einer TIA Portal Openness API können Sie mittels Programmcode Objekte auf Gleichheit prüfen:

- Sie prüfen dabei mit dem Operator `==`, ob zwei Objektreferenzen gleich sind.
- Mit der Methode `System.Object.Equals()` prüfen Sie, ob beide Objekte in Bezug auf das TIA Portal wirklich identisch sind.

Programmcode

Um auf Objektreferenztypen zu prüfen, ändern Sie den folgenden Programmcode:

```
...
//Composition
DeviceComposition sameCompA = project.Devices;
DeviceComposition sameCompB = project.Devices;
if (sameCompA.Equals(sameCompB))
{
    Console.WriteLine("sameCompA is equal to sameCompB");
}
if (!(sameCompA == sameCompB))
{
    Console.WriteLine("sameCompA is not reference equal to sameCompB");
}
DeviceComposition sameCompAsA = sameCompA;
if (sameCompAsA.Equals(sameCompA))
{
    Console.WriteLine("sameCompAsA is equal to sameCompA");
}
if (sameCompAsA == sameCompA)
{
    Console.WriteLine("sameCompAsA is reference equal to sameCompA");
}
MultiLingualGraphicComposition notSameComp = project.Graphics;
if (!sameCompA.Equals(notSameComp))
{
    Console.WriteLine("sameCompA is not equal to notSameComp");
}
```


5.2.19 Lese-Operationen für Attribute

Gruppenoperationen und Standardleseoperationen für Attribute

TIA Portal Openness unterstützt den Zugriff auf Attribute über die folgenden Methoden, die auf Objektebene verfügbar sind:

- Gruppenoperation für Lesezugriff
- Standardleseoperationen

Programmcode für Gruppenoperationen

```
//Exercise GetAttributes and GetAttributeNames
//get all available attributes for a device,
//then get the names for those attributes, then display the results.
private static void DynamicTest(Project project)
{
    Device device = project.Devices[0];
    IList<string> attributeNames = new List<string>();
    IList<EngineeringAttributeInfo> attributes =
    ((IEngineeringObject)device).GetAttributeInfos();
    foreach (EngineeringAttributeInfo engineeringAttributeInfo in attributes)
    {
        string name = engineeringAttributeInfo.Name;
        attributeNames.Add(name);
    }
    IList<object> values = ((IEngineeringObject)device).GetAttributes(attributeNames);
    for (int i = 0; i < attributes.Count; i++)
    {
        Console.WriteLine("attribute name: " + attributeNames[i] + " value: " + values[i]);
    }
}
```

Gruppenoperation für Lesezugriff

Diese Methode ist für jedes Objekt verfügbar:

```
public abstract IList<object> GetAttributes(IEnumerable<string>
names);
```

Standardleseoperationen

Die folgenden Operationen sind verfügbar:

- Namen der verfügbaren Attribute abrufen:
Verwenden Sie die Methode `GetAttributeInfos()` (Seite 126) auf einem `IEngineeringObject`.
- Generische Methode zum Lesen eines Attributs

```
public abstract object GetAttribute(string name);
```

Hinweis

Dynamische Attribute werden in IntelliSense nicht angezeigt, weil ihre Verfügbarkeit vom Status der Objektinstanz abhängig ist.

5.2.20 Transaktionsbehandlung

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Operation

Eine innerhalb eines zugehörigen TIA Portal-Prozesses geöffnete Persistenz (Projekt, Bibliothek usw.) kann durch eine TIA Portal Openness-Client-Anwendung geändert werden. Sie können diese Änderung durch eine einzelne Operation oder durch eine Reihe von Operationen herbeiführen. Während der Aktivität ist es sinnvoll, diese Operationen in einer einzelnen Undo-Einheit zu gruppieren, um logischere Arbeitsabläufe zu erhalten. Darüber hinaus ergeben sich durch das Gruppieren von Operationen in einer einzelnen Undo-Einheit auch Leistungsvorteile. Um dies zu unterstützen, bietet die Klasse "ExclusiveAccess" die Methode "Transaction(ITransactionSupport persistence, string undoDescription)". Der Aufruf aus dieser Methode führt zur Instanziierung eines neuen beendbaren Objekts des Typs "Transaction". Sie müssen eine Beschreibung des Inhalts der Transaktion liefern (das Textattribut darf nicht Null oder leer sein). Solange diese Instanz nicht beendet wurde, werden alle Operationen von Client-Anwendungen in einer einzelnen Undo-Einheit innerhalb des zugehörigen TIA Portal-Prozesses gruppiert.

Ändern Sie den folgenden Programmcode, um eine Instanz des Typs "Transaction" zu erfassen:

```
ExclusiveAccess exclusiveAccess = ...;  
Project project = ...;  
using (Transaction transaction = exclusiveAccess.Transaction(project, "My Operation"))  
{  
    ...  
}
```

Hinweis

Verwenden Sie zum Instanzieren einer "Transaction" eine "using"-Anweisung. So stellen Sie sicher, dass die Transaktion selbst beim Auftreten von Ausnahmen ordnungsgemäß beendet wird, um so die Transaktion rückgängig zu machen.

Konsistente Umsetzung oder Rückgängigmachung

Mit Hilfe einer "Transaction" in einer Client-Anwendung stellen Sie sicher, dass es einen vorhersehbaren Weg gibt, einen Satz von Änderungen wirksam werden zu lassen oder rückgängig zu machen. Ihre Client-Anwendung muss entscheiden, ob Änderungen an einer Persistenz wirksam werden sollen oder nicht. Hierzu muss Ihre Anwendung anfordern, dass die Änderungen im Geltungsbereich einer geöffneten Transaktion wirksam werden, wenn die Transaktion durch Aufrufen der Methode "Transaction.CommitOnDispose()" beendet wird. Wenn diese Methode im Codeablauf nie aufgerufen wird, werden die Änderungen im Geltungsbereich der geöffneten Transaktion bei Beendigung der Transaktion automatisch rückgängig gemacht.

Hinweis

Jeder Aufruf von Transaction.CommitOnDispose(), der nach Auftreten einer Siemens.Engineering.TargetInvocationException innerhalb einer offenen Transaktion erfolgt, führt dazu, dass eine wiederherstellbare Ausnahme ausgelöst wird.

Wenn nach Stellung dieser Anforderung eine Ausnahme auftritt, werden alle Änderungen im Geltungsbereich einer geöffneten Transaktion bei Beendigung der Transaktion dennoch rückgängig gemacht.

Zwei Eigenschaften sind verfügbar, mit deren Hilfe der 'Commit'-Status einer Transaktion festgestellt werden kann:

Eigenschaft	Datentyp	Beschreibung
Transaction.CanCommit	Boolescher Wert	Gibt an, ob ein vorheriger Aufruf von Transaction.CommitOnDispose() erfolgreich ausgeführt wird oder ein darauffolgender Aufruf von Transaction.CommitOnDispose() zu einer wiederherstellbaren Ausnahme führt
Transaction.CommitRequested	Boolescher Wert	Gibt an, ob ein Aufruf von Transaction.CommitOnDispose() durch den Benutzer innerhalb der gegebenen Transaktion erfolgt ist

Ändern Sie den folgenden Programmcode, um eine einzelne Undo-Einheit in dem angehängten TIA Portal mit zwei "Create"-Änderungen zu erzeugen:

```
ExclusiveAccess exclusiveAccess = ...;
Project project = ...;
// Create a single undo unit with name "My Operation" in attached TIA Portal
using (Transaction transaction = exclusiveAccess.Transaction(project, "My Operation"))
{
    project.DeviceGroups.Create("My Group 1");
    project.DeviceGroups.Create("My Group 2");
    transaction.CommitOnDispose();
}
```

Ändern Sie folgenden Programmcode, um keine Undo-Einheit zu erzeugen:

```
ExclusiveAccess exclusiveAccess = ...;
Project project = ...;
using (Transaction transaction = exclusiveAccess.Transaction(project, "My Operation"))
{
    project.DeviceGroups.Create("My Folder 1");
    project.DeviceGroups.Create("My Folder 2");
}
```

Ändern Sie folgenden Programmcode, um im angehängten TIA Portal keine Undo-Einheit zu erzeugen. Dies ist auf eine beim zweiten Versuch, den Ordner "My Folder 1" zu erstellen, ausgelöste Ausnahme zurückzuführen – obwohl die Anforderung CommitOnDispose vor der Auslösung der Ausnahme gestellt wurde:

```
ExclusiveAccess exclusiveAccess = ...;
Project project = ...;
using (Transaction transaction = exclusiveAccess.Transaction(project, "My Operation"))
{
    project.DeviceGroups.Create("My Folder 1");
    project.DeviceGroups.Create("My Folder 2");
    transaction.CommitOnDispose();
    project.DeviceGroups.Create("My Folder 1");
}
```

Ändern Sie folgenden Programmcode, um im angehängten TIA Portal keine Undo-Einheit zu erzeugen. Dies ist auf eine beim zweiten Versuch, den Ordner "My Folder 1" zu erstellen, ausgelöste Ausnahme zurückzuführen – obwohl der Try/Catch-Baustein den Aufruf umgibt, der die ausgelöste Ausnahme verursacht hat:

```
ExclusiveAccess exclusiveAccess = ...;
Project project = ...;
using (Transaction transaction = exclusiveAccess.Transaction(project, "My Operation"))
{
    project.DeviceGroups.Create("My Folder 1");
    project.DeviceGroups.Create("My Folder 2");
    transaction.CommitOnDispose();
    try
    {
        project.DeviceGroups.Create("My Folder 1");
    }
    catch (Siemens.Engineering.TargetInvocationException ex)
    {
        ...
    }
}
```

Ändern Sie folgenden Programmcode, um eine im angehängten TIA Portal erstellte Undo-Einheit zu erzeugen. Dies ist auf den Aufruf von Transaction.CommitOnDispose() nach einer vorher im Codestream ausgelösten Siemens.Engineering.TargetInvocationException

zurückzuführen – obwohl ein Try/Catch-Baustein den Aufruf umgibt, der die ausgelöste Ausnahme verursacht hat:

```
ExclusiveAccess exclusiveAccess = ...;
Project project = ...;
using (Transaction transaction = exclusiveAccess.Transaction(project, "My Operation"))
{
    try
    {
        project.DeviceGroups.Create("My Folder 1");
        project.DeviceGroups.Create("My Folder 1");
    }
    catch (Siemens.Engineering.TargetInvocationException ex)
    {
        ...
    }
    transaction.CommitOnDispose();
}
```

Ändern Sie folgenden Programmcode, um eine Transaktion zurückzusetzen. Dies ist darauf zurückzuführen, dass `Transaction.CommitOnDispose()` nie ausgeführt wird, wenn die Ausführung einer vorherigen Anweisung im Rahmen der angegebenen Transaktion eine Ausnahme auslöst.

```
ExclusiveAccess exclusiveAccess = ...;
Project project = ...;
try
{
    using (Transaction transaction = exclusiveAccess.Transaction(project, "My Operation"))
    {
        project.DeviceGroups.Create("My Folder 1");
        project.DeviceGroups.Create("My Folder 1");
        transaction.CommitOnDispose();
    }
}
catch (Exception ex)
{
    ...
}
```

Einschränkungen

Folgende Aktionen sind innerhalb einer Transaktion nicht zulässig. Der Aufruf dieser Aktionen führt zu einer wiederherstellbaren Ausnahme:

- Compile
- Go Online
- Go Offline
- ProjectText Import
- ProjectText Export

- Open Global Library
- Close Global Library
- Project Create
- Project Open
- Project OpenWithUpgrade
- Project Save
- Project SaveAs
- Project Close

Undo-Verhalten

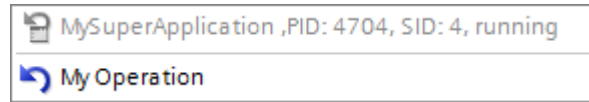
Von einer TIA Portal Openness-Client-Anwendung ausgeführte Aktionen können in Undo-Einheiten innerhalb des angehängten TIA Portal-Prozesses resultieren. Jeder dieser Undo-Einträge wird unter einem Ortseintrag gruppiert. Dieser Ortseintrag setzt sich aus den folgenden Informationen aus der Client-Anwendung zusammen:

- der Assembly-Titel aus den Manifestdaten, wenn verfügbar, andernfalls der Prozessname
- die Prozess-ID
- die SID (Openness-Session-ID der TIA-Portal-Instanz der Client-Anwendung)
- optional ein Hinweis darauf, dass der Client-Prozess noch läuft

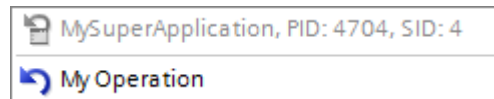
Diese Einträge gehören zu einer der folgenden beiden Arten:

1. Die Operationen, die als Resultat der Verwendung einer "Transaction" in einer Undo-Transaktion zusammengefasst werden, haben die Beschreibung wie von der Client-Anwendung bereitgestellt, als die "Transaction" instanziiert wurde.

- Undo-Eintrag für eine laufende Client-Anwendung:

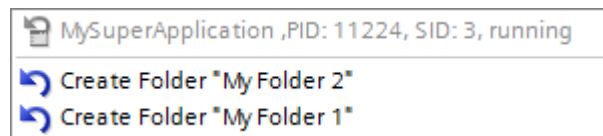


- Undo-Eintrag für eine gestoppte Client-Anwendung:

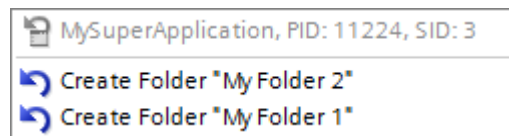


2. Für die Operationen, die individuell ausgeführt werden, existieren individuelle Undo-Einträge zur Beschreibung der Operation, wie im entsprechenden Metadatenbefehl definiert.

- Undo-Eintrag für eine laufende Client-Anwendung:



- Undo-Eintrag für eine gestoppte Client-Anwendung:



5.2.21 Ein Objekt DirectoryInfo/FileInfo erstellen

Anwendung

Die Instanzen der Klassen `DirectoryInfo` und `FileInfo` müssen einen absoluten Pfad enthalten. Andernfalls führen die Methoden, die die Objekte `DirectoryInfo` oder `FileInfo` verwenden, zu einer Ausnahme.

Programmcode

Um ein Objekt `DirectoryInfo` oder `FileInfo` zu erstellen, ändern Sie den folgenden Programmcode:


```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;

using Siemens.Engineering.SW.ExternalSources;

using Siemens.Engineering.SW.Tags;

using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
namespace CreateObjectDirectoryInfoFileInfo
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            RunTiaPortal();
            //..
            //Create a DirectoryInfo object
            string directoryPath = @"D:\Test\Project 1";
            DirectoryInfo directoryInfo = new DirectoryInfo(directoryPath);
            //Create a FileInfo object
            //please adapt the path and the extension apx to the installed version of TIA Portal
            string fileName = @"D:\Test\Project 1\Project 1.apx";
            FileInfo fileInfo = new FileInfo(fileName);
            //...
        }
        private static void RunTiaPortal()
        {
            Console.WriteLine("Starting TIA Portal");
            using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
            {
                Console.WriteLine("TIA Portal has started");
                ProjectComposition projects = tiaPortal.Projects;
                Console.WriteLine("Opening Project...");
                // please adapt the path and the extension apx to the installed version of TIA Portal
                FileInfo projectPath = new FileInfo(@"C:\Demo\AnyCompanyProject.apx");
                //edit the path according to your project
                Project project = null;
            }
        }
    }
}
```

```
try
{
project = projects.OpenWithUpgrade(projectPath);
}
catch (Exception)
{
Console.WriteLine(String.Format("Could not open project {0}", projectPath.FullName));
Console.WriteLine("Demo complete hit enter to exit");
Console.ReadLine();
return;
}
Console.WriteLine(String.Format("Project {0} is open", project.Path.FullName));
IterateThroughDevices(project);
project.Close();
Console.WriteLine("Demo complete hit enter to exit");
Console.ReadLine();
}
}
private static void IterateThroughDevices(Project project)
{
if (project == null)
{
Console.WriteLine("Project cannot be null");
return;
}
Console.WriteLine(String.Format("Iterate through {0} device(s)", project.Devices.Count));
foreach (Device device in project.Devices)

{
Console.WriteLine(String.Format("Device: \"{0}\".", device.Name));
}
Console.WriteLine();
}
}
}
```

5.2.22 Unterstützung der Selbstbeschreibung für Attribute, Navigatoren, Aktionen und Dienste

Verwendung

In TIA Portal Openness beschreibt jeder `IEngineeringServiceProvider` der TIA Portal Openness API seine Fähigkeiten für potentielle Aufrufe.

Unterstützung der Selbstbeschreibung für IEngineeringObject

Methodenname	Rückgabewerte
GetCompositionInfos	Gibt eine Sammlung von Objekten des Typs EngineeringCompositionInfo zurück, die die verschiedenen Zusammensetzungen dieser Objekte beschreiben. EngineeringCompositionInfo wird nachstehend beschrieben.
GetAttributeInfos	Gibt eine Sammlung von Objekten des Typs EngineeringAttributeInfo zurück, die die verschiedenen Attribute dieser Objekte beschreiben. EngineeringAttributeInfo wird nachstehend beschrieben.
GetInvocationInfos	Gibt eine Sammlung von Objekten des Typs EngineeringInvocationInfo zurück, die die verschiedenen Aktionen dieser Objekte beschreiben. EngineeringInvocationInfo wird nachstehend beschrieben.

Unterstützung der Selbstbeschreibung für IEngineeringServiceProvider

Methodenname	Rückgabewerte
GetServiceInfos	Gibt eine Sammlung von Objekten des Typs EngineeringServiceInfo zurück, die die verschiedenen Dienste dieser Objekte beschreiben. EngineeringServiceInfo wird nachstehend beschrieben.

Unterstützung der Selbstbeschreibung für IEngineeringComposition

Methodenname	Rückgabewerte
GetCreationInfos	Gibt eine Sammlung von Objekten des Typs EngineeringCreationInfo zurück, die die verschiedenen Aktionen dieser Objekte beschreiben. EngineeringcreationInfo wird nachstehend beschrieben.

Klasse EngineeringCompositionInfo

Attributname	Rückgabewerte
Name	Name der Zusammensetzung

Klasse EngineeringAttributeInfo

Attributname	Rückgabewerte
AccessMode	Die vom Attribut unterstützte Zugriffsebene. Dieses Attribut ist kombinierbar und wird nachstehend ausführlich beschrieben.
Name	Name des Attributs.
CreateRelevance	Ebene, bis zu der das Attribut relevant für das Erstellen ist.
SupportedTypes	Die potenziell unterstützten Typen des Attributs.

Klasse EngineeringInvocationInfo

Attributname	Rückgabewerte
Name	Name der Aktion.
ParameterInfos	Eine Sammlung von Objekten des Typs EngineeringInvocationParameterInfo, die von der Aktion möglicherweise benötigte Parameter beschreiben. EngineeringInvocationParameterInfo wird nachstehend beschrieben.

Klasse EngineeringServiceInfo

Attributname	Rückgabewerte
Type	Der Diensttyp als System.Type-Objekt.

Enum AccessMode

Enumerationswert	Rückgabewerte
None	Keine gültige Option.
Read	Das Attribut kann gelesen werden.
Write	Das Attribut kann geschrieben werden.

Klasse EngineeringInvocationParameterInfo

Attributname	Rückgabewerte
Name	Name des Parameters
Type	Typ des Parameters als System.Type-Objekt

Klasse EngineeringCreationInfo

Attributname	Rückgabewerte
Type	Die Eigenschaft Type gibt ein Objekt System.Type zurück.
ParameterInfos	Die Eigenschaft ParameterInfos gibt eine Sammlung von Objekten EngineeringCreationParameterInfo zurück.

Klasse EngineeringCreationParameterInfo

Attributname	Rückgabewerte
Name	Name des betreffenden Parameters

Programmcode

AccessMode ist eine Merker-Enumeration, ihre Werte können wie der folgende Programmcode kombiniert werden:

```
EngineeringAttributeAccessMode value = EngineeringAttributeAccessMode.Read |  
EngineeringAttributeAccessMode.Write;
```

Ändern Sie den folgenden Programmcode, um alle Attribute eines IEngineeringObject zu suchen und Änderungen am Zugriffsmodus dieser Attribute vorzunehmen.

```
...
IEngineeringObject engineeringObject = ...;
IList<EngineeringAttributeInfo> attributeInfos = engineeringObject.GetAttributeInfos();
foreach(EngineeringAttributeInfo attributeInfo in attributeInfos)
{
    switch (attributeInfo.AccessMode)
    {
        case EngineeringAttributeAccessMode.Read:
            ...
            break;
        case EngineeringAttributeAccessMode.Write:
            ...
            break;
        case EngineeringAttributeAccessMode.Read|EngineeringAttributeAccessMode.Write:
            ...
            break;
    }
}
...
```

Ändern Sie folgenden Programmcode dafür, wie Create und GetCreationInfos gemeinsam verwendet werden sollen.

```
...
Project project = tiaPortal.Projects.Open(projectPath);
IEngineeringComposition deviceGroupComposition = project.DeviceGroups;
EngineeringCreationInfo deviceGroupCreationInfo = deviceGroupComposition.GetCreationInfos()
[0];
EngineeringCreationParameterInfo deviceGroupCreationParameterInfo =
deviceGroupCreationInfo.ParameterInfos[0];
string deviceGroupCreationParameterName = deviceGroupCreationParameterInfo.Name;
string groupName = "Example";
IDictionary<string, object> createParameters = new Dictionary<string, object>
{
    {deviceGroupCreationParameterName, groupName }
};
IEngineeringObject group = deviceGroupComposition.Create(deviceGroupCreationInfo.Type,
createParameters);
...
```

5.2.23 Attribute von Bausteinen, DBs und UDTs festlegen

Einleitung

Es ist möglich, (allgemeine) Attribute von (in einer beliebigen Programmiersprache geschriebenen) Bausteinen, von DBs und von UDTs über die Openness API zu schreiben und zu lesen.

Technologieobjekte sind intern DBs, deshalb gilt diese Funktion auch für TOs.

Die gleichen Prüfregeln sind auch auf den Namen anzuwenden, und zwar mittels SetAttribute der Openness API für Bausteine, DBs (TOs) und UDTs, die auf die GUI angewendet werden.

Bei einer Regelverletzung wird eine entsprechende Anwenderausnahme zurückgegeben und der Attributwert wird nicht geändert.

Diese Erweiterung der API betrifft nicht den XML-Export/-Import.

Allgemeine Attribute

Legende

- X: Das Attribut ist relevant, Schreiben in XML, Zugriff über API möglich
- -: Das Attribut ist nicht zugänglich, das Attribut existiert nicht

Attribute	Datentyp	Verfügbarkeit über die API		
		CodeBlock	DB	UDT
AutoNumber	Bool	X (XML); RW (API)	X (XML); RW (API)	-
CodeModifiedDate	DateTime	X (XML); R (API)	X (XML); R (API)	-
CompileDate	DateTime	X (XML); R (API)	X (XML); R (API)	-
CreationDate	DateTime	X (XML); R (API)	X (XML); R (API)	X (XML); R (API)
HeaderAuthor	String	X (XML); R (API)	X (XML); R (API)	-
HeaderFamily	String	X (XML); R (API)	X (XML); R (API)	-
HeaderName	String	X (XML); R (API)	X (XML); R (API)	-
HeaderVersion	System.Version	X (XML); R (API)	X (XML); R (API)	-
InstanceOfName	String	-	X (XML); R (API)	-
Schnittstelle	String	X (XML); - (API)	X (XML); - (API)	X (XML); - (API)
InterfaceModifiedDate	DateTime	X (XML); R (API)	X (XML); R (API)	X (XML); R (API)
IsConsistent	Bool	X (XML); R (API)	X (XML); R (API)	X (XML); R (API)
IsKnowHowProtected	Bool	X (XML); R (API)	X (XML); R (API)	X (XML); R (API)
MemoryLayout (bisher: Optimization)	MemoryLayout	X (XML); R (API)	X (XML); R (API)	-
ModifiedDate	DateTime	X (XML); R (API)	X (XML); R (API)	X (XML); R (API)
Name	String	X (XML); RW (API)	X (XML); RW (API)	X (XML); RW (API)
Number	Int32	X (XML); RW (API)	X (XML); RW (API)	-
ParameterModified	DateTime	X (XML); R (API)	X (XML); R (API)	-
ProgrammingLanguage	ProgrammingLanguage	X (XML); R (API)	X (XML); R (API)	-
SecondaryType	String	X (XML); R (API)	-	-
StructureModified	DateTime	X (XML); R (API)	X (XML); R (API)	-

5.2.24 Anmerkungen zur Leistung von TIA Portal Openness

Stammobjekt

In Importdateien können Sie mehrere Stammobjekte angeben.

Beispiel: Statt einer Textliste für jede XML-Datei können Sie mehrere Textlisten in einer XML-Datei erzeugen.

TIA Portal Openness-Funktionen

Der erste Aufruf einer TIA Portal Openness-Funktion kann länger dauern als nachfolgende Aufrufe der TIA Portal Openness-Funktion.

Beispiel: Wenn Sie nacheinander mehrere Exporte von Konfigurationsdaten durchführen, kann der erste Export länger dauern als die nachfolgenden Exporte.

5.3 Funktionen für Projekte und Projektdaten

5.3.1 Projekt öffnen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Das zu öffnende Projekt ist in keiner anderen Instanz des TIA Portals geöffnet.

Hinweis

Rückgängigmachen eines Projekt-Upgrades

Wenn Sie das Upgrade eines Projekts auf V14 SP1 rückgängig machen, nachdem Sie das Projekt mit TIA Portal Openness verbunden haben, treten Konflikte auf.

Verwendung

Zum Öffnen eines Projekts verwenden Sie die Methode `Projects.Open`. Geben Sie in der Methode `Projects.Open` einen Pfad zu dem gewünschten Projekt ein.

Die Methode `Projects.Open` greift ausschließlich auf Projekte zu, die mit der aktuellen Version des TIA Portals angelegt oder auf die aktuelle Version hochgerüstet wurden. Wenn Sie mit der Methode `Projects.Open` auf ein Projekt einer Vorgängerversion zugreifen,

wird eine Ausnahme zurückgegeben. Verwenden Sie die Methode `OpenWithUpgrade` zum Öffnen von Projekten, die mit älteren Versionen von TIA Portal angelegt wurden.

Hinweis**Kein Zugriff auf schreibgeschützte Projekte**

TIA Portal Openness kann nur auf Projekte mit Lese- und Schreibrechten zugreifen.

Programmcode

Um ein Projekt zu öffnen, ändern Sie den folgenden Programmcode:

```
Project project = tiaPortal.Projects.Open(new FileInfo(@"D:\Project_1\Project_1.apXX"));
if (project != null)
{
    try
    {
        ...
    }
    finally
    {
        project.Close();
    }
}
```

Öffnen eines UMAC-geschützten Projekts

Sie können auch ein UMAC-geschütztes Projekt öffnen. Die Überlast der Funktion `Open` benötigt einen zusätzlichen Parameter vom Typ `UmacDelegate`. Dieser zusätzliche Parameter ermöglicht es dem Aufrufer, einen Handler anzugeben, der während der UMAC-Authentifizierung verwendet werden soll. Der neue `UmacDelegate` wird mit einer Methode implementiert, die einen Parameter vom Typ `UmacCredentials` enthält. `UmacCredentials` hat zwei Eigenschaften, 'Name' vom Typ `String` und 'Type' vom Typ `UmacUserType` sowie eine Methode `SetPassword` mit einem Parameter vom Typ `SecureString`. Durch die Verwendung von `UmacUserType.Project` wird auf einen UMAC-Projektumfang hingewiesen, während durch die Verwendung von `UmacUserType.Global` auf einen UMAC-Anwendungsumfang hingewiesen wird (d.h. gesteuert von einem UMAC-Server).

Programmcode

```
...
Siemens.Engineering.Project project = tiaPortal.Projects.Open(new
FileInfo(@"D:\Project_3\Project_2.apXX"), MyUmacDelegate);
if (project != null)
{
try
{
...
}
finally
{
project.Close();
}
}
...
private static void MyUmacDelegate(UmacCredentials umacCredentials)
{
SecureString password = ...; // Get password from a secure location
umacCredentials.Type = UmacUserType.Project;
umacCredentials.Name = "SomeUser";
umacCredentials.SetPassword(password);
}
...
}
```

Öffnen eines UMAC-geschützten Projekts mit einer neuen Authentifizierung

Vor TIA Portal Openness V17 kann das geschützte Projekt durch Eingabe des Benutzernamens und Passworts über UMAC Delegate in Open Project API geöffnet werden.

Ab TIA Portal Openness V17 hat UMAC neue Authentifizierungsmechanismen eingeführt, mit denen Sie ein geschütztes Projekt öffnen können. Die neuen Authentifizierungsmechanismen sind Desktop Single Sign-On (SSO) und Anonymous User.

TIA Portal Openness V17 unterstützt die folgenden Authentifizierungsmechanismen. Die Implementierung von Openness ist ferner unabhängig von den Einstellungen im TIA Portal.

- Desktop Single Sign-On (SSO)
- Anonymous User
- Interaktive Anmeldung
- Anmeldedaten (aus TIA Portal V15.1)

Zur Unterstützung der zusätzlichen Authentifizierungsmethoden wurde in Openness ein neuer Ereignisname "Authentication" im TIA Portal-Objekt eingeführt. Die Änderungen sind als Teil von V17 Engineering realisiert. Ereignis "Authentication" wird ausgeführt, sofern es

vom Benutzer im Openness-Skript registriert ist, wenn das Öffnen eines Projekts aufgerufen wird. Ereignis "Authentication" wird nur beim Öffnen eines geschützten Projekts ausgeführt.

```
var tiaPortal = new TiaPortal();
tiaPortal.Authentication += OnAuthentication;
private static void OnAuthentication(object sender, AuthenticationEventArgs e)
{
    e.AuthenticationTypeProvider = AuthenticationTypeProvider.Credentials;
}
```

Sie können die zusätzlichen Authentifizierungsmethoden mit den Ereignisargumenten `AuthenticationEventArgs` verwenden. Mit `AuthenticationTypeProvider` wird die Authentifizierungsart bereitgestellt. Die Anmeldungsart wird anhand des obigen Beispiels eingestellt.

Die Werte für `AuthenticationTypeProvider` sind nachfolgend angegeben.

- `AuthenticationTypeProvider.DesktopSso`: Projekt öffnen mit Single-Sign-On-Benutzer, es wird kein Passwort verlangt
- `AuthenticationTypeProvider.Anonymous`: Projekt öffnen mit Anonymous User, es wird kein Passwort verlangt
- `AuthenticationTypeProvider.Interactive`: Projekt öffnen mit interaktiver Anmeldung, Benutzername und Passwort müssen über die Benutzeroberfläche eingegeben werden
- `AuthenticationTypeProvider.Credentials`: Projekt mit einem Benutzernamen und Passwort öffnen.

Das Ereignis gilt für das Öffnen des Projekts in einer Multiuser-Umgebung. Die oben beschriebenen Authentifizierungsmethoden gelten, wenn ein geschütztes Projekt in einer lokalen Sitzung oder in der Serversicht geöffnet wird.

Wenn Ausnahmen vorliegen, zum Beispiel für eine Single-Sign-On-Sitzung oder bei einem ungültigen Benutzernamen / Passwort, wird eine `EngineeringTargetInvocationException` generiert.

Mehrere Projekte öffnen

Sie können in einer Instanz des TIA Portals ein primäres Projekt und mehrere sekundäre Projekte gleichzeitig öffnen. Sie können ein Projekt als primäres oder sekundäres Projekt öffnen. Wird ein Projekt als primäres Projekt geöffnet, wird es in der Projektnavigation angezeigt, wenn die Openness-Anwendung mit dem TIA Portal verknüpft ist. Beim Öffnen eines Projekts als sekundäres Projekt wird das Projekt nicht auf der Benutzeroberfläche dargestellt. Sekundäre Projekte werden immer schreibgeschützt geöffnet. Ein Anwender mit Lese- und Schreibrechten für UMAC-geschützte Projekte hat nur Leserechte, wenn das Projekt als sekundäres Projekt geöffnet wird. Es ist nicht notwendig, dass ein primäres Projekt geöffnet ist, damit ein sekundäres Projekt geöffnet werden kann.

Jedes geöffnete Projekt kann mit Hilfe der Projektzusammensetzung in der Instanz des TIA Portals enumeriert werden. Die Reihenfolge der Projekte in der Zusammensetzung wird durch die Reihenfolge, in der sie geöffnet wurden, bestimmt. Wird ein Projekt geschlossen, wird der Index aller Projekte neu berechnet.

Programmcode

```
TiaPortal tiaPortal = ...;
Project project1 = tiaPortal.Projects.Open(new
FileInfo(@"D:\Project_1\Project_1.apXX"), null, ProjectOpenMode.Primary);
Project project3 = tiaPortal.Projects.Open(new
FileInfo(@"D:\Project_3\Project_3.apXX"), null, ProjectOpenMode.Secondary);
bool isPrimary = project3.IsPrimary
```

Mit älteren Versionen angelegte Projekte öffnen

Verwenden Sie die Methode `OpenWithUpgrade` zum Öffnen eines Projekts, das mit der älteren Version des TIA Portals angelegt wurde. Die Methode legt ein neues, aktualisiertes Projekt an und öffnet es.

Wenn Sie auf ein mit einer älteren Version erstelltes Projekt zugreifen, wird eine Ausnahme zurückgegeben.

Hinweis

Wenn Sie auf ein mit der aktuellen Version erstelltes Projekt zugreifen, wird das Projekt geöffnet.

Programmcode

Ändern Sie den folgenden Programmcode zum Öffnen eines Projekts über die Methode `OpenWithUpgrade`:

```
Project project = tiaPortal.Projects.OpenWithUpgrade(new
FileInfo(@"D:\Some\Path\Here\Project.apXX"));
if (project != null)
{
    try
    {
        ...
    }
    finally
    {
        project.Close();
    }
}
```

Programmcode für ein UMAC-geschütztes Projekt

Sie können auch ein UMAC-geschütztes Projekt öffnen, das mit einer Vorgängerversion von TIA Portal angelegt wurde. Eine Überlastfunktion von `OpenWithUpgrade` benötigt einen zusätzlichen Parameter vom Typ `UmacDelegate`. `OpenWithUpgrade` wird auch bei sekundären Projekten unterstützt.

```
...
    Siemens.Engineering.Project project = tiaPortal.Projects.OpenWithUpgrade(new
FileInfo(@"D:\Project_1\Project.apXX"), MyUmacDelegate);
...
```

5.3.2 Projekt anlegen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Siehe Verbindung zum TIA Portal aufbauen (Seite 84)

Verwendung

Projekte können über die TIA Portal Openness API erstellt werden

- durch Aufrufen der Methode Create auf ProjectComposition
- durch Aufrufen der Methode Create auf IEngineeringComposition

ProjectComposition.Create

Ändern Sie folgenden Programmcode:

```
TiaPortal tiaPortal = ...;
ProjectComposition projectComposition = tiaPortal.Projects;
DirectoryInfo targetDirectory = new DirectoryInfo(@"D:\TiaProjects");
// Create a project with name MyProject
Project project = projectComposition.Create(targetDirectory, "MyProject");
```

Entsprechend diesem Beispiel

- wird ein Ordner "D:\TiaProjects\MyProject" erzeugt.
- wird eine Projektdatei "D:\TiaProjects\MyProject\MyProject.apXX" erstellt.

Hinweis

Parameter targetDirectory

Der Parameter targetDirectory kann auch einen UNC (Universal Naming Convention)-Pfad darstellen, von daher kann ein Projekt auch auf einem freigegebenen Laufwerk im Netzwerk erstellt werden.

IEngineeringComposition.Create

Ändern Sie folgenden Programmcode:

```
TiaPortal tiaPortal = ...;
ProjectComposition projectComposition = tiaPortal.Projects;

//allows the user to give optional create parameters like author, comment in addition to
mandatory create parameters (targetdirectory, projectname)

IEnumerable<KeyValuePair<string, object>> createParameters = new [] {
    new KeyValuePair<string, object>("TargetDirectory", new
DirectoryInfo(@"D:\TiaProjects")), // Mandatory
    new KeyValuePair<string, object>("Name", "MyProject"), // Mandatory
    new KeyValuePair<string, object>("Author", "Bob"), // Optional
    new KeyValuePair<string, object>("Comment", "This project was created with
Openness") // Optional };

// Create a project with both mandatory and optional parameters
((IEngineeringComposition)projectComposition).Create(typeof (Project), createParameters);
```

Entsprechend diesem Beispiel:

- wird ein Ordner "D:\TiaProjects\MyProject" erzeugt.
- wird eine Projektdatei "D:\TiaProjects\MyProject\MyProject.aPXX" mit den Projektattributen Autor gleich "Bob" und Kommentar gleich "This project was created with openness" erstellt.

Parameter zum Erstellen eines Projekts mit optionalen Projektattributen

Parameter	Datentyp	Obligatorisch	Beschreibung
Author	String	Nein	Autor eines Projekts.
Comment	String	Nein	Kommentar zum Projekt.
Name	String	Ja	Name eines Projekts.
TargetDirectory	DirectoryInfo	Ja	Verzeichnis, in dem der erstellte Projektordner enthalten ist.

5.3.3 Auf allgemeine Einstellungen des TIA Portals zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Über TIA Portal Openness können Sie auf allgemeine Einstellungen des TIA Portals zugreifen:

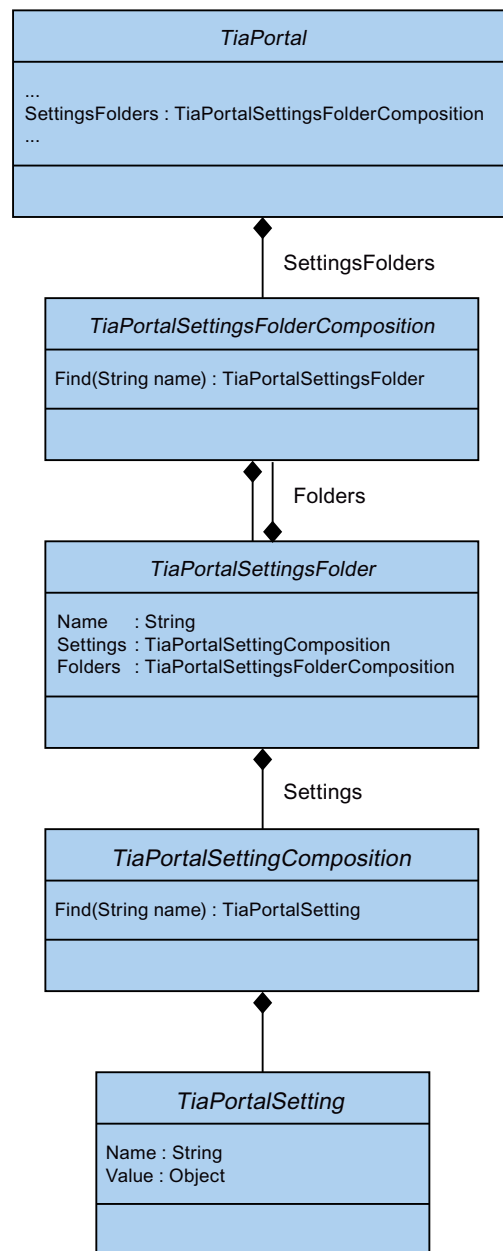
- Aktuelle Sprache der Benutzeroberfläche
- Option "In Projekt suchen", um den Suchindex zu erstellen, der für die Suche in einem Projekt benötigt wird.

Die folgende Tabelle zeigt die Details der zugänglichen Einstellungen im Abschnitt "Allgemein" der Einstellungen im TIA Portal. Die Instanz `TiaPortalSettingsFolder` hat den Namen "Allgemein".

Name	Datentyp	Schreibbar	Beschreibung
"SearchInProject"	System.Boolean	r/w	Aktiviert oder deaktiviert die Erstellung des Suchindex zum Suchen innerhalb eines Projekts.
"UserInterfaceLanguage"	System.CultureInfo	r/w	Zeigt die aktive Sprache der Benutzeroberfläche des TIA Portals oder die Angabe der aktiven Sprache der Benutzeroberfläche an.

Der Zugriff auf diese Einstellungen erfolgt über die Klasse `TiaPortalSettingsFolder`. Die Klasse `TiaPortalSettingsFolder` ist über das Attribut `Settings` der Klasse `TiaPortal` zugänglich.

Die folgende Abbildung zeigt die spezifischen Einstellungen in TIA Portal Openness:



Programmcode: Im Projekt suchen

Um die Option "In Projekt suchen" zu aktivieren/deaktivieren, ändern Sie den folgenden Programmcode.

```
private static void SetSearchInProject(Project project)
{
    TiaPortalSettingsFolder generalSettingsFolder =
    tiaPortal.SettingsFolders.Find("General");
    TiaPortalSetting searchSetting =
    generalSettingsFolder.Settings.Find("SearchInProject");

    if (((bool)searchSetting.Value))
    {
        searchSetting.Value = false;
    }
}
```

Programmcode: Sprache der Benutzeroberfläche

Um auf die aktuelle Sprache der Benutzeroberfläche zuzugreifen, ändern Sie den folgenden Programmcode:

```
private static void SetUILanguage(Project project)
{
    TiaPortalSettingsFolder generalSettingsFolder =
    tiaPortal.SettingsFolders.Find("General");

    TiaPortalSetting UILanguageSetting =
    generalSettingsFolder.Settings.Find("UserInterfaceLanguage");

    if (((CultureInfo)UILanguageSetting.Value) != CultureInfo.GetCultureInfo("de-DE"))
    {
        UILanguageSetting.Value = CultureInfo.GetCultureInfo("de-DE");
    }
}
```

Siehe auch

Projekt öffnen (Seite 131)

5.3.4 Projekt archivieren und abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Ein Projekt ist gespeichert.
Siehe Projekt speichern (Seite 160)

Verwendung

Mit dem TIA Portal Openness können Sie den Zustand des geöffneten und gespeicherten Projekts vor dem Durchführen weiterer Änderungen archivieren und später das archivierte Projekt abrufen.

Projekt archivieren

Sie können ein TIA Portal-Projekt mit der am Objekt `Siemens.Engineering.Project` verfügbaren TIA Openness API archivieren.

```
public void Archive(System.IO.DirectoryInfo targetDirectory, string targetName,  
Siemens.Engineering.ProjectArchivationMode archivationMode)
```

'targetName' ist der Name der erstellten Datei, unabhängig davon, ob sie archiviert oder nicht archiviert ist. Diese Datei kann Dateierweiterungen oder keine Dateierweiterungen enthalten. Für den Wert `ProjectArchivationMode` als `Compressed` und `DiscardRestorableDataAndCompressed`, lautet der archivierte Dateiname wie von Ihnen angegeben. Wenn Sie keine Erweiterung oder außer "zap15_1" bzw. "zap14" usw. keine Erweiterung angeben, kann dieses archivierte Projekt nicht außerhalb der Openness API vom TIA Portal abgerufen werden. Wenn Sie weder `Compressed` noch `DiscardRestorableDataAndCompressed` wählen, ist das Ergebnis die Standarddateistruktur eines Projekts der aktuellen Version.

Hinweis

Sie müssen das Projekt vor dem Aufrufen der Archivierungs-API speichern. Falls das Projekt nicht gespeicherte Änderungen enthält, löst das Archiv eine `EngineeringTargetInvocationException` aus.

ProjectArchivationMode

Die Enumeration **ProjectArchivationMode** hat vier Werte.

ProjectArchivationMode	Beschreibung
None	- Mit den Originaldateien werden keine speziellen Aktionen durchgeführt. Der Modus ähnelt dem Vorgang "Speichern unter". - In diesem Modus wird keine komprimierte ZIP-Datei erstellt. - Der Unterschied zu SaveAs besteht in diesem Fall darin, dass das Archiv den persistenten Speicherort nicht auf den neuen Ordner Archived umstellt, was jedoch bei SaveAs der Fall ist.
DiscardRestorableData	- Bei der Dateispeicherung wird das Projekt in einer internen Datendatei abgelegt. Diese Datei wird anschließend bei jeder Änderung an den Projektdaten größer. Beim Modus DiscardRestorableData wird diese Datendatei reorganisiert (dabei wird nur die neueste Version der Objekte gespeichert und der Verlauf aus der Datei gelöscht). Zwischenzeitlich angefallene Daten, die Dateien des IM-Verzeichnisses und des Verzeichnisses tmp werden nicht an den Archivspeicherort kopiert. - In diesem Modus wird keine komprimierte ZIP-Datei erstellt.
Compressed	Die vom Archivierungsprozess erstellte Struktur des TMP-Projektordners wird zu einem ZIP-kompatiblen Archiv komprimiert. Nach Erstellung der ZIP-Datei wird die TMP-Ordnerstruktur gelöscht.
DiscardRestorableDataAndCompressed	In der vom Archivierungsprozess erstellten Struktur des TMP-Projektordners werden die wiederherstellbaren Daten verworfen und dann zu einem ZIP-kompatiblen Archiv komprimiert. Nach Erstellung der ZIP-Datei wird die TMP-Ordnerstruktur gelöscht.

Programmcode: Projekt archivieren

Ändern Sie den folgenden Programmcode, um ohne Erweiterung auf das Archivprojekt zuzugreifen:

```
public void ArchivingProjectWithoutExtension()
{
    var tiaPortal = new TiaPortal(TiaPortalMode.WithoutUserInterface);
    var projectFilePath = @"E:\Sample1\Sample1.ap15_1";
    var project = tiaPortal.Projects.Open(new FileInfo(projectFilePath));
    var archiveDirectory = @"E:\Archive";
    project.Archive(new DirectoryInfo(archiveDirectory), "ArchiveProjectNameWithoutExtension",
        ProjectArchivationMode.Compressed);
}
```

Ändern Sie den folgenden Programmcode, um mit der Erweiterung '.archive' auf das Archivprojekt zuzugreifen:

```
public void ArchivingProjectWithExtension()
{
    var tiaPortal = new TiaPortal(TiaPortalMode.WithoutUserInterface);
    var projectFilePath = @"E:\Sample1\Sample1.ap15_1";
    var project = tiaPortal.Projects.Open(new FileInfo(projectFilePath));
    var archiveDirectory = @"E:\Archive";
    project.Archive(new DirectoryInfo(archiveDirectory), "ArchiveProjectName.archive",
        ProjectArchivationMode.Compressed);
}
```

Hinweis

Sie können jede beliebige Erweiterung verwenden. Das Verhalten des Archivs ist mit und ohne Erweiterung identisch.

Projekt abrufen

Über die Openness API können Sie ein archiviertes TIA Portal-Projekt abrufen. Der Abruf ist nur mit einem komprimierten Archiv möglich. Bei archivierten Projekten mit 'ProjectArchivationMode.None' oder 'ProjectArchivationMode.DiscardRestorableData' kann der Enumerationswert nicht von dieser API abgerufen werden. Diese API ist am Objekt "Siemens.Engineering.ProjectComposition" verfügbar.

```
public Siemens.Engineering.Project Retrieve(System.IO.FileInfo sourcePath,
    System.IO.DirectoryInfo targetDirectory)
```

Zum Abrufen eines UMAC-geschützten Projekts steht zum Abfragen der Anmeldedaten eine überlastete API mit UmacDelegate bereit. Dann wird die folgende Definition verwendet:

```
public Siemens.Engineering.Project Retrieve(System.IO.FileInfo sourcePath,
    System.IO.DirectoryInfo targetDirectory, Siemens.Engineering.UmacDelegate umacDelegate)
```

Beim Abrufen wird außerdem ProjectOpenMode unterstützt, für das "Primary" oder "Secondary" angegeben werden kann. Dann wird die folgende API-Definition verwendet:

```
public Siemens.Engineering.Project Retrieve(System.IO.FileInfo sourcePath,
    System.IO.DirectoryInfo targetDirectory, Siemens.Engineering.UmacDelegate umacDelegate,
    Siemens.Engineering.ProjectOpenMode projectOpenMode)
```

Hinweis

UmacDelegate kann als null übergeben werden, wenn das Projekt nicht geschützt ist.

5.3 Funktionen für Projekte und Projektdaten

Wenn das archivierte Projekt aus einer früheren TIA Portal-Version stammt, müssen Sie die `RetrieveWithUpgrade` API, aufrufen. Dann wird die folgende API-Definition verwendet:

```
public Siemens.Engineering.Project RetrieveWithUpgrade(System.IO.FileInfo sourcePath,  
System.IO.DirectoryInfo targetDirectory);
```

Wenn das archivierte Projekt aus einer früheren TIA Portal-Version stammt und UMAC-geschützt ist, steht eine andere überlastete API mit `UmacDelegate` zur Verfügung. Dann wird die folgende API-Definition verwendet:

```
public Siemens.Engineering.Project RetrieveWithUpgrade(System.IO.FileInfo sourcePath,  
System.IO.DirectoryInfo targetDirectory, Siemens.Engineering.UmacDelegate umacDelegate)
```

Wenn das archivierte Projekt aus einer früheren TIA Portal-Version stammt und auch der Projektöffnungsmodus angegeben werden muss, wird die folgende API-Definition verwendet:

```
public Siemens.Engineering.Project RetrieveWithUpgrade(System.IO.FileInfo sourcePath,  
System.IO.DirectoryInfo targetDirectory, Siemens.Engineering.UmacDelegate umacDelegate,  
Siemens.Engineering.ProjectOpenMode projectOpenMode)
```

Hinweis

`UmacDelegate` kann als null übergeben werden, wenn das Projekt nicht geschützt ist. Wenn das Projekt geschützt ist, sind die Benutzerdaten eines Administrators erforderlich, um die Aktion `RetrieveWithUpgrade` durchzuführen.

Programmcode: Projekt abrufen

Um auf das abzurufende Projekt zuzugreifen, ändern Sie folgenden Programmcode:

```
public void RetrieveProject()  
{  
    var archivePath = @"E:\Archive\Sample1.zap15_1";  
    var retrievedProjectsDirectory = @"E:\RetrievedProjects";  
    var tiaPortal = new TiaPortal(TiaPortalMode.WithoutUserInterface);  
    tiaPortal.Projects.Retrieve(new FileInfo(archivePath), new  
        DirectoryInfo(retrievedProjectsDirectory));  
}
```

Siehe auch

[Verbindung zum TIA Portal aufbauen \(Seite 84\)](#)

[Projekt öffnen \(Seite 131\)](#)

[Projekt speichern \(Seite 160\)](#)

5.3.5 Schreibgeschütztes TIA Portal-Projekt öffnen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Mithilfe von TIA Portal Openness können auch bei einem schreibgeschützten TIA Portal-Projekt bestimmte Vorgänge ausgeführt werden. Sie können auf ein schreibgeschütztes Projekt zugreifen, können jedoch nicht den vollständigen Funktionsumfang nutzen, der einem Benutzer mit Lese-/Schreibzugriff zur Verfügung steht. Beispielsweise kann ein Benutzer, der nur über Leserechte verfügt, mit Openness ein UMAC-geschütztes Projekt wie in Öffnen eines Projekts (Seite 131) beschrieben öffnen. Die Nutzbarkeit dieser Funktion gilt jedoch nicht für Referenzprojekte.

Die Liste der Openness-Funktionen, die Ihnen beim Zugriff auf ein schreibgeschütztes Projekt zur Verfügung stehen, lässt sich in zwei Gruppen unterteilen: in integrierte Aktionen und in aktivierte nicht-verändernde Aktionen.

Integrierte Funktionen

- GetAttribute(s) oder Verwendung der Abruffunktion für ein beliebiges Attribut eines beliebigen Objekts
- GetComposition bei einem Objekt, auf das Zugriff möglich ist
- GetService bei einem Objekt, auf das Zugriff möglich ist
- Find-Aktionen bei einem Objekt, auf das Zugriff möglich ist
- Navigation bei einem Objekt, auf das Zugriff möglich ist
- Feststellen des Vorhandenseins von Objekten, auf die Zugriff möglich ist, und Zugriff auf diese Objekte in Zusammensetzungen und Zuordnungen
- System.Object-Methoden bei einem Objekt, auf das Zugriff möglich ist

Aktivierte nicht-verändernde Aktionen

- Project.Close (...)
- PlcBlock.ShowInEditor ()
- CaxProvider.Export (Device,...)
- CaxProvider.Export (Project,...)

Siehe auch

Projekt öffnen (Seite 131)

5.3.6 Auf Sprachen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

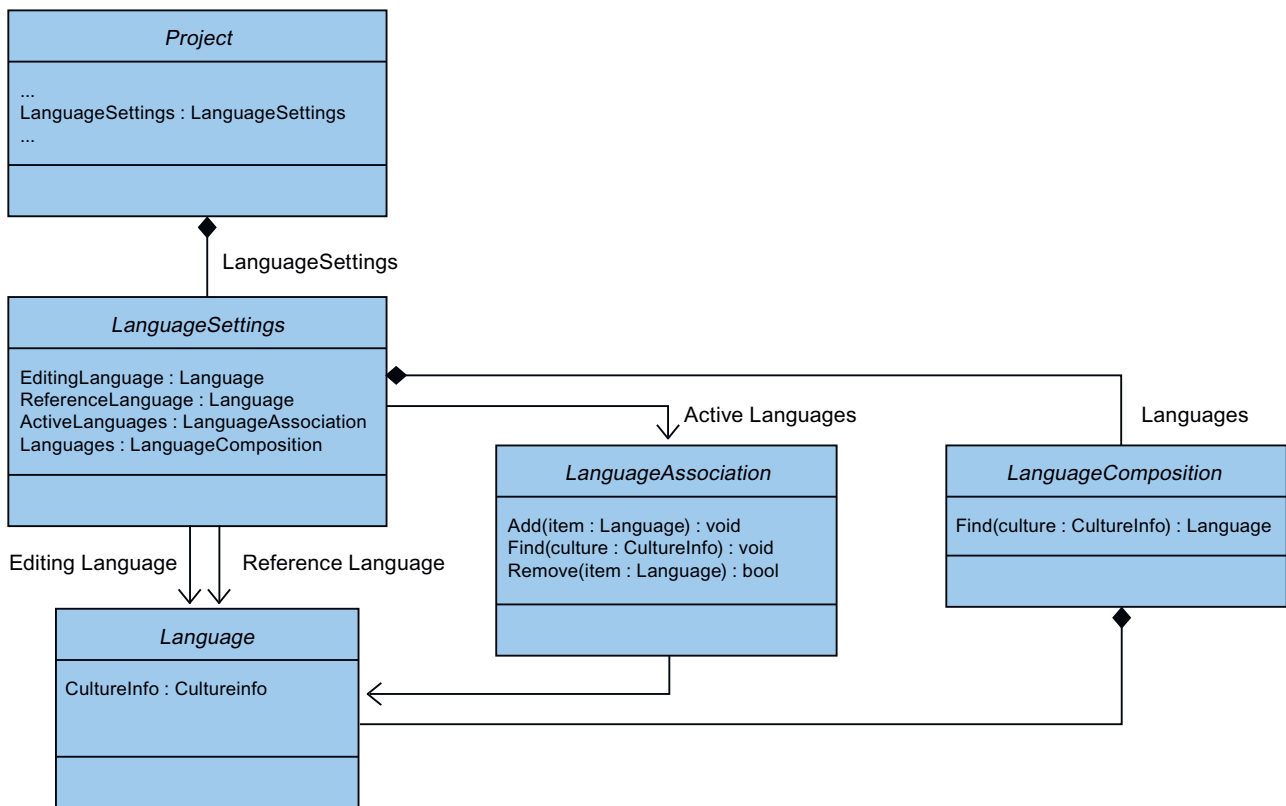
Verwendung

Im TIA Portal können Sie die Projektsprache im Editor "Projektsprachen" festlegen und verwalten.

TIA Portal Openness unterstützt den folgenden Zugriff auf die Projektsprachen:

- Iteration durch unterstützte Sprachen.
- Suchen in der Sammlung unterstützter Sprachen mit Hilfe von `System.Globalization.CultureInfo`.
- Zugriff auf einzelne Sprachen. Jedes Sprachobjekt enthält ein einziges schreibgeschütztes Attribut `Culture` vom Typ `System.Globalization.CultureInfo`.
- Zugriff auf eine Sammlung aktiver Sprachen.
- Suchen in der Sammlung aktiver Sprachen über mit Hilfe von `System.Globalization.CultureInfo`.
- Hinzufügen einer Sprache zu einer Sammlung aktiver Sprachen.
- Entfernen einer Sprache aus einer Sammlung aktiver Sprachen.
- Festlegen einer Bearbeitungssprache.
- Festlegen einer Referenzsprache.

Die Funktionalitäten werden vom Objekt `LanguageSettings` bereitgestellt. Die folgende Abbildung zeigt das Modell in TIA Portal Openness:



Programmcode: Sprachen festlegen

Ändern Sie den folgenden Programmcode, um eine Sprache festzulegen. Wenn Sie über TIA Portal Openness eine inaktive Sprache festlegen, wird die Sprache der Sammlung aktiver Sprachen hinzugefügt.

```

public void SettingLanguages()
{
    Project project = ...;
    LanguageSettings languageSettings = project.LanguageSettings;
    LanguageComposition supportedLanguages = languageSettings.Languages;
    LanguageAssociation activeLanguages = languageSettings.ActiveLanguages;
    Language supportedGermanLanguage = supportedLanguages.Find(CultureInfo.GetCultureInfo("de-DE"));
    activeLanguages.Add(supportedGermanLanguage);
    languageSettings.EditingLanguage = supportedGermanLanguage;
    languageSettings.ReferenceLanguage = supportedGermanLanguage;
}

```

Programmcode: Eine aktive Sprache deaktivieren

Um eine aktive Sprache zu deaktivieren, ändern Sie den folgenden Programmcode. Wenn Sie eine Sprache deaktivieren, die als Referenz- oder Bearbeitungssprache verwendet wird, ist die ausgewählte Sprache konsistent mit dem Verhalten in der Benutzeroberfläche.

```
public void DeactivatingLanguages()  
{  
    Project project = ...;  
    LanguageSettings languageSettings = project.LanguageSettings;  
    LanguageAssociation activeLanguages = languageSettings.ActiveLanguages;  
    Language activeGermanLanguage = activeLanguages.Find(CultureInfo.GetCultureInfo("de-DE"));  
    activeLanguages.Remove(activeGermanLanguage);  
}
```

Siehe auch

Hierarchie von Hardware-Objekten des Objektmodells (Seite 67)

5.3.7 Objektstruktur und -Attribute ermitteln

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die Navigationsstruktur in der Objekthierarchie können Sie mit der Schnittstelle `IEngineeringObject` ermitteln. Das Ergebnis wird als Liste zurückgegeben:

- Untergeordnete Objekte
- Kindzusammensetzungen
- Alle Attribute

Signatur

Verwenden Sie zum Ermitteln von Attributen die Methode `GetAttributeInfos`.

```
IList<EngineeringAttributeInfo>  
IEngineeringObject.GetAttributeInfos();
```


Programmcode: Objekte oder Zusammensetzungen ermitteln

Um alle Namen von Zusammensetzungen anzuzeigen, verwenden Sie den folgenden Programmcode:

```
public static void DisplayCompositionInfos(IEngineeringObject obj)
{
    IList<EngineeringCompositionInfo> compositionInfos = obj.GetCompositionInfos();
    foreach (EngineeringCompositionInfo compositionInfo in compositionInfos)
    {
        Console.WriteLine(compositionInfo.Name);
    }
}
```

Wenn Sie den Rückgabewert kennen, ändern Sie den folgenden Programmcode:

```
public static DeviceItemComposition GetDeviceItemComposition(Device device)
{
    IEngineeringCompositionOrObject composition = ((IEngineeringObject)
    device).GetComposition("DeviceItems");
    DeviceItemComposition deviceItemComposition = (DeviceItemComposition)composition;
    return deviceItemComposition;
}
```

Programmcode: Attribute ermitteln

Um Attribute eines Objekts mit bestimmten Zugriffsrechten in einer Liste zurückzugeben, ändern Sie folgenden Programmcode:

```
public static void DisplayAttributenInfos(IEngineeringObject obj)
{
    IList<EngineeringAttributeInfo> attributeInfos = obj.GetAttributeInfos();
    foreach (EngineeringAttributeInfo attributeInfo in attributeInfos)
    {
        Console.WriteLine("Attribute: {0} - AccessMode {1} ", attributeInfo.Name,
            attributeInfo.AccessMode);
        switch (attributeInfo.AccessMode)
        {
            case EngineeringAttributeAccessMode.Read: Console.WriteLine("Attribute: {0} - Read
                Access", attributeInfo.Name);
                break;
            case EngineeringAttributeAccessMode.Write: Console.WriteLine("Attribute: {0} - Write
                Access", attributeInfo.Name);
                break;
            case EngineeringAttributeAccessMode.Read | EngineeringAttributeAccessMode.Write:
                Console.WriteLine("Attribute: {0} - Read and Write Access",
                    attributeInfo.Name);
                break;
        }
    }
}

public static string GetNameAttribute(IEngineeringObject obj)
{
    Object nameAttribute = obj.GetAttribute("Name");
    return (string)nameAttribute;
}
```

5.3.8 Auf Software-Ziel zugreifen**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie den folgenden Programmcode, um ein Software-Ziel verfügbar zu machen:

```
public void AccessSoftwareTarget()
{
    SoftwareContainer softwareContainer =
        ((IEngineeringServiceProvider)deviceItem).GetService<SoftwareContainer>();
    if (softwareContainer != null)
    {
        Software software = softwareContainer.Software;
    }
}
```

Ändern Sie den folgenden Programmcode, um auf die Software-Attribute zuzugreifen:

```
SoftwareContainer softwareContainer =
    ((IEngineeringServiceProvider)deviceItem).GetService<SoftwareContainer>();
if (softwareContainer != null)
{
    PlcSoftware software = softwareContainer.Software as PlcSoftware;
    string name = software.Name;
}
```

5.3.9 Auf mehrsprachige Texte zugreifen und sie enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Mehrsprachige Texte im TIA Portal sind beispielsweise Project.Comment, PlcTag.Comment usw. In TIA Portal Openness werden die mehrsprachigen Texte vom Objekt `MultilingualText` dargestellt. Ein Objekt `MultilingualText` besteht aus `MultilingualTextItemComposition`.

`MultilingualTextItemComposition` unterstützt die folgende Find-Methode:

- `Find(<language: Siemens.Engineering.Language>):MultilingualTextItem`

Jedes `MultilingualTextItem` hat die folgenden Attribute:

Attributname	Datentyp	Schreibbar	Beschreibung
Language	Siemens.Engineering.Language	r/o	Sprache dieses Elements
Text	System.String	r/w	Für diese Sprache angegebener Text

Programmcode: Mehrsprachige Texte festlegen

```
...
    Language englishLanguage = project.LanguageSettings.Languages.Find(new CultureInfo("en-US"));
    MultilingualText comment = project.Comment;
    MultilingualTextItemComposition mltItemComposition = comment.Items;
    MultilingualTextItem englishComment = mltItemComposition.Find(englishLanguage);
    englishComment.Text = "English comment";
...
```

Programmcode: Mehrsprachige Texte für Geräte festlegen

Ändern Sie folgenden Programmcode, um mehrsprachige Texte für Geräte und Geräteelemente festzulegen:

```
...
    var mltObject = device.GetAttribute("CommentML");
    MultilingualText multilingualText = mltObject as MultilingualText;
    if (multilingualText != null)
    {
        Language englishLanguage = project.LanguageSettings.Languages.Find(new
CultureInfo("en-US"));
        MultilingualTextItem multilingualTextItem =
multilingualText.Items.Find(englishLanguage);
        if (multilingualTextItem != null)
        {
            multilingualTextItem.Text = comment;
        }
    }
...
```

5.3.10 Projekteigenschaften aktualisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Eigenschaft "Simulationsunterstützung" eines Projekts aktualisieren, um ein PLC-Programm zu erstellen. Die Einstellung für die Eigenschaft "Simulationsunterstützung" ist für die virtuelle SINUMERIK-Steuerung und mit PLCSIM Advanced simulierte SIMATIC-PLCs verfügbar.

Openness unterstützt das folgende Attribut zum Zugriff auf die Eigenschaft "Simulationsunterstützung" eines TIA Portal-Projekts. Die API ist am Objekt "Siemens.Engineering.Project" verfügbar.

Attributname	Datentyp	Zugriff	Beschreibung
IsSimulationDuringBlockCompilationEnabled	System.Boolean	Lesen / Schreiben	Gibt an, ob die Unterstützung für die Simulation bei der Bausteinübersetzung für das Projekt aktiviert ist.
IsVirtualPlcDuringBlockCompilationEnabled	System.Boolean	Lesen / Schreiben	Gibt an, ob die Unterstützung für die virtuelle PLC bei der Bausteinübersetzung für das Projekt aktiviert ist.

Programmcode

Ändern und verwenden Sie den folgenden Programmcode, um auf den Attributwert zuzugreifen:

```
Project project = ...;
// Read the attribute value
var attributeValue = project.IsSimulationDuringBlockCompilationEnabled;
var attributeValue = project.IsVirtualPlcDuringBlockCompilationEnabled;
//Write the attribute value to false
project.IsSimulationDuringBlockCompilationEnabled = false;
//Write the attribute value to True
project.IsVirtualPlcDuringBlockCompilationEnabled = true;
```

Siehe auch

Projekt öffnen (Seite 131)

5.3.11 Projektbezogene Attribute lesen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Mit dieser Funktion können Sie projektbezogene Attribute aus der TIA Portal Openness API erhalten. Die gelieferten Informationen umfassen Projektattribute, Projekthistorie und vom Projekt genutzte Produkte.

Projektattribute

Die Projektattribute liefern die folgenden Informationen:

Attributname	Datentyp	Schreibbar	Beschreibung
Author	System.String	r/o	Autor des Projekts
Comment	Siemens.Engineering.MultilingualText	r/o	Kommentar des Projekts
Copyright	System.String	r/o	Copyright-Hinweis des Projekts
CreationTime	System.DateTime	r/o	Zeitpunkt, zu dem das Projekt angelegt wurde
Family	System.String	r/o	Familie des Projekts
IsModified	System.Boolean	r/o	Gibt wahr aus, wenn das Projekt geändert wurde
LanguageSettings	Siemens.Engineering.LanguageSettings	r/o	Handhabt Projektsprachen
LastModified	System.DateTime	r/o	Zeitpunkt, zu dem das Projekt zuletzt geändert wurde
LastModifiedBy	System.String	r/o	Autor der letzten Änderung
Name	System.String	r/o	Name des Projekts
Path	System.IO.FileInfo	r/o	Absoluter Pfad des Projekts
Size	System.Int64	r/o	Größe des Projekts in KB
Version	System.String	r/o	Version des Projekts

Ändern Sie den folgenden Programmcode, um auf projektbezogene Attribute zuzugreifen:

```
Project project = ...;
string author = project.Author;
string name = project.Name;
string path = project.Path;
DateTime creationTime = project.CreationTime;
DateTime modificationTime = project.LastModified;
string lastModifiedBy = project.LastModifiedBy;
string version = project.Version;
MultilingualText comment = project.Comment;
string copyright = project.Copyright;
string family = project.Family;
Int64 size = project.Size;
LanguageSettings languageSettings = project.LanguageSettings;
```

Ändern Sie das folgende Programm, um die Projektsprachen zu enumerieren:

```
Project project = ...;
LanguageComposition languages = project.LanguageSettings.Languages;
foreach (Language language in languages)
{
    CultureInfo lang = language.Culture;
}
```

Ändern Sie den folgenden Programmcode, um Kommentartext zu erhalten:

```
Project project = ...;
Language english =
project.LanguageSettings.ActiveLanguages.Find(CultureInfo.GetCultureInfo("en-US"));

MultilingualText projectComment = project.Comment;
MultilingualTextItem textItem = project.Comment.Items.Find(english);
string text = textItem.Text;
```

Projekthistorie

Die Projekthistorie ist eine Zusammensetzung von Objekten des Typs `HistoryEntry`, die folgende Informationen enthalten:

Attributname	Datentyp	Schreibbar	Beschreibung
Text	System.String	r/o	Ereignisbeschreibung
DateTime	System.DateTime	r/o	Zeitpunkt, zu dem das Ereignis aufgetreten ist

5.3 Funktionen für Projekte und Projektdaten

Ändern Sie den folgenden Programmcode, um HistoryEntries zu enumerieren und auf deren Attribute zuzugreifen:

```
Project project = ...;
HistoryEntryComposition historyEntryComposition = project.HistoryEntries;
foreach (HistoryEntry historyEntry in historyEntryComposition)
{
    string entryText = historyEntry.Text;
    DateTime entryTime = historyEntry.DateTime;
}
```

Hinweis

Das Textattribut von HistoryEntry enthält einen String in der gleichen Sprache wie die Benutzeroberfläche. Wenn eine TIA Portal Openness-Anwendung an ein TIA Portal ohne Benutzeroberfläche angehängt wird, liegt der String immer auf Englisch vor.

Verwendete Produkte

Das Objekt UsedProduct enthält die folgenden Informationen:

Attributname	Datentyp	Schreibbar	Beschreibung
Name	System.String	r/o	Name des verwendeten Projekts
Version	System.String	r/o	Version des Produkts

Ändern Sie den folgenden Programmcode, um UsedProduct zu enumerieren und auf die Attribute zuzugreifen.

```
Project project = ...;
UsedProductComposition usedProductComposition = project.UsedProducts;
foreach (UsedProduct usedProduct in usedProductComposition)
{
    string productName = usedProduct.Name;
    string productVersion = usedProduct.Version;
}
```

5.3.12 Projektgrafik löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie den folgenden Programmcode, um eine Grafiksammlung zu löschen:

```
//Deletes a single project graphic entry
public static void DeletesSingleProjectGraphicEntry(Project project)
{
    MultiLingualGraphicComposition graphicsAggregation = project.Graphics;
    MultiLingualGraphic graphic = graphicsAggregation.Find("Graphic XYZ");
    graphic.Delete();
}
```

5.3.13 Projekt übersetzen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Alle Geräte sind "offline".

Verwendung

Die API-Schnittstelle unterstützt die Übersetzung von Geräten und Programmbausteinen. Das Übersetzungsergebnis wird als Objekt zurückgegeben. In Abhängigkeit vom Objekttyp wird die HW- oder SW- oder HW/SW-Übersetzung zur Verfügung gestellt. Die folgenden Objekttypen werden unterstützt:

- Device - HW & SW
- Device mit fehlersicherer CPU - SW mit ausgeschalteter F-Aktivierung
- DeviceItem - HW
- CodeBlock - SW
- DataBlock - SW
- HmiTarget - SW
- PlcSoftware - SW
- PlcType - SW
- PlcBlockSystemGroup - SW
- PlcBlockUserGroup - SW
- PlcTypeSystemGroup - SW
- PlcTypeUserGroup - SW

Hinweis**Format des Zeitstempels**

Alle Zeitstempel sind in UTC. Wenn Sie die lokale Uhrzeit anzeigen möchten, können Sie `DateTime.ToLocalTime()` verwenden.

Hinweis

Wurde dem Sicherheitsprogramm ein Passwort zugewiesen (Seite 1235), so müssen Sie sich im Sicherheitsprogramm anmelden (Seite 1237), um die Softwareänderungen zu übersetzen. Sonst wird der Vorgang abgebrochen und eine Ausnahme ausgelöst.

Signatur

Verwenden Sie für die Übersetzung die Methode `ICompilable`.

```
ICompilable compileService =  
iEngineeringServiceProvider.GetService<ICompilable>();  
  
CompilerResult result = compileService.Compile();
```

Hinweis

Alle Geräte müssen vor dem Start der Übersetzung "offline" sein.

Programmcode

Um die Softwareänderungen eines Objekts vom Typ `HmiTarget` zu übersetzen, ändern Sie folgenden Programmcode:

```
public static void CompileHmiTarget(HmiTarget hmiTarget)  
{  
    ICompilable compileService = hmiTarget.GetService<ICompilable>();  
    CompilerResult result = compileService.Compile();  
}
```

Um die Softwareänderungen eines Objekts vom Typ `PlcSoftware` zu übersetzen, ändern Sie folgenden Programmcode:

```
public static void CompilePlcSoftware(PlcSoftware plcSoftware)  
{  
    ICompilable compileService = plcSoftware.GetService<ICompilable>();  
    CompilerResult result = compileService.Compile();  
}
```

Um die Softwareänderungen eines Objekts vom Typ `CodeBlock` zu übersetzen, ändern Sie folgenden Programmcode:

```
public static void CompileCodeBlock(PlcSoftware plcSoftware)
{
    CodeBlock block = plcSoftware.BlockGroup.Blocks.Find("MyCodeBlock") as CodeBlock;
    if (block != null)
    {
        ICompilable compileService = block.GetService<ICompilable>();
        CompilerResult result = compileService.Compile();
    }
}
```

Um das Übersetzungsergebnis auszuwerten, ändern Sie folgenden Programmcode:

```
private void WriteCompilerResults(CompilerResult result)
{
    Console.WriteLine("State:" + result.State);
    Console.WriteLine("Warning Count:" + result.WarningCount);
    Console.WriteLine("Error Count:" + result.ErrorCount);
    RecursivelyWriteMessages(result.Messages);
}

private void RecursivelyWriteMessages(CompilerResultMessageComposition messages, string indent = "")
{
    indent += "\t";
    foreach (CompilerResultMessage message in messages)
    {
        Console.WriteLine(indent + "Path: " + message.Path);
        Console.WriteLine(indent + "DateTime: " + message.DateTime);
        Console.WriteLine(indent + "State: " + message.State);
        Console.WriteLine(indent + "Description: " + message.Description);
        Console.WriteLine(indent + "Warning Count: " + message.WarningCount);
        Console.WriteLine(indent + "Error Count: " + message.ErrorCount);
        RecursivelyWriteMessages(message.Messages, indent);
    }
}
```

5.3.14 Ein Projekt in UMAC ändern

Einleitung

Sie können mit TIA Portal Openness die Benutzerverwaltung und Zugriffsrechte für ein Projekt ändern. Mit diesem Funktionsrecht kann bei einem geschützten Projekt "Projekt über Openness API ändern" einer Rolle zugewiesen werden, die wiederum einem Benutzer zugewiesen werden kann.

Wenn Sie dieses Funktionsrecht in einem geschützten Projekt besitzen / nicht besitzen, ändern sich bestimmte Verhaltensweisen in TIA Portal Openness, wenn ein geschütztes Projekt mit bestimmten Anmeldedaten geöffnet wird. Die Änderungen in der Openness

API gelten nur für geschützte Projekte und Projekte, die mit ProjectOpenMode als primäres Projekt geöffnet werden. ProjectOpenMode als sekundäres Projekt wird nicht berücksichtigt.

Geändertes Verhalten für Funktionsrechte

Das nachfolgend beschriebene Verhalten der Openness Funktionsrechte gilt für alle Versionen der Siemens.Engineering.dll (V15, V15.1, V16 und V17), wenn diese mit TIA Portal V17 laufen.

- Wenn Sie das Funktionsrecht "Projekt über Openness API ändern" zusammen mit Lese-Schreibrechten für das Projekt besitzen, haben Sie vollen Zugriff auf die Openness API und können jeden Vorgang mit der Openness API ausführen.
- Wenn Sie kein Funktionsrecht "Projekt über Openness API ändern" zusammen mit Lese-Schreibrechten / Leserechten für das Projekt besitzen, dürfen Sie nur schreibgeschützte Vorgänge in der Openness API ausführen. Schreibgeschützte Vorgänge sind beispielsweise Export, Navigieren zu einem Gerät usw. Wenn Sie dann versuchen, einen Vorgang wie z. B. einen Import auszuführen, der datenorientiert ist (Änderung von Projektdaten im TIA Portal), wird abhängig von den Funktionsrechten, die Sie besitzen, eine Ausnahme zurückgegeben.
- Bei Openness APIs, die auf die Benutzeroberfläche bezogen sind (z. B. ShowInEditor), hängen die Funktionsrechte nicht von "Projekt über Openness API ändern" ab, sondern von dem Funktionsrecht für die Benutzeroberfläche, das als solches behandelt wird. Die folgenden APIs sind Ausnahmen von Punkt 1 und 2. Die Openness API übergeht das Funktionsrecht "Projekt über Openness API ändern" und die APIs sind zulässig unabhängig davon, ob das Funktionsrecht "Projekt über Openness API ändern" gesetzt / nicht gesetzt ist.
 - project.ShowHwEditor(View.Network)
 - project.ShowHwEditor(View.Topology)
 - device.ShowInEditor(View.Device)
 - plcBlock.ShowInEditor()
 - plcType.ShowInEditor()
 - plcTagTable.ShowInEditor()
 - plcForceTable.ShowInEditor()
 - plcWatchTable.ShowInEditor()

5.3.15 Projekt speichern

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Ein Projekt speichern

- Zum Speichern eines Projekts verwenden Sie die Methode `Save()`.
- Um ein Projekt mit einem anderen Namen oder in einem anderen Verzeichnis zu speichern, verwenden Sie die Methode `SaveAs()`.

Programmcode

Ändern Sie folgenden Programmcode, um ein Projekt zu öffnen und zu speichern:

```
public static void SaveProject(TiaPortal tiaPortal)
{
    Project project = ...;
    //Use the code in the try block to open and save a project
    try
    {
        //please adapt the path and the extension apx to the installed version of TIA Portal
        project = tiaPortal.Projects.Open(new FileInfo(@"Some\Path\MyProject.apx"));
        //begin of code for further implementation
        //...
        //end of code'
        project.Save();
    }
    //Use the code in the final block to close a project
    finally
    {
        if (project != null)
            project.Close();
    }
}
```

Ändern Sie folgenden Programmcode, um ein Projekt mit einem anderen Namen oder an einem anderen Ort zu speichern:

```
...
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
FileInfo fileInfoExistingProject = new FileInfo(@"D:\SampleProjects\SampleProject.apXX");
DirectoryInfo dirInfoSaveAsProject = new
DirectoryInfo(@"D:\SampleProjects\SampleProjectSaveAs");
Project sampleProject = portal.Projects.Open(fileInfoExistingProject);
sampleProject.SaveAs(dirInfoSaveAsProject);
...
```

5.3.16 Projekt schließen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um ein Projekt zu schließen, ändern Sie folgenden Programmcode:

```
public static void CloseProject(Project project)
{
    project.Close();
}
```

5.3.17 Export/Import von Systemdiagnoseeinstellungen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Systemdiagnoseeinstellungen für ein TIA Portal-Projekt exportieren und importieren. Das Format für Export und Import ist eine .dat-Datei.

Zum Exportieren und Importieren der Einstellungen für die Systemdiagnose werden die folgenden Dienste und Klassen verwendet.

Name	Beschreibung
SystemdiagnosticsSettingsDataProvider	Dienstklasse für die Bereitstellung einer Methode für Export und Import
SystemdiagnosticsSettingsExportImportResults	Rückgabe des Ergebnisses des Export- oder Importvorgangs

Die SystemdiagnosticsSettingsExportImportResultState ENUM hat folgende Werte:

ENUM	Beschreibung	Wert
SystemdiagnosticsSettingsExportImportResults	Gibt als Ergebnis zurück, ob die Datei erfolgreich gespeichert/Einstellungen geändert wurde(n) oder nicht.	Error
		Success
		Warning

Programmcode: Einstellungen für Systemdiagnose exportieren

Ändern Sie folgenden Programmcode, um die Einstellungen für Systemdiagnosen zu exportieren:

```
public static void ExportSystemDiagnosticSettings(project tiaProject)
{
    Siemens.Engineering.HW.Systemdiagnostics.Settings.SystemdiagnosticsSettingsDataProvider
    settingsProvider =
    tiaProject.GetService<Siemens.Engineering.HW.Systemdiagnostics.Settings.SystemdiagnosticsS
    ettingsDataProvider>();
    Siemens.Engineering.HW.Systemdiagnostics.Settings.SystemdiagnosticsSettingsExportImportRes
    ult exportResult = settingsProvider.Export(new FileInfo(@"D:\Temp\SysDiagSettings.dat"));
    Siemens.Engineering.HW.Systemdiagnostics.Settings.SystemdiagnosticsSettingsExportImportRes
    ultState resultState = exportResult.State;
}
```

Programmcode: Einstellungen für Systemdiagnose importieren

Ändern Sie folgenden Programmcode, um die Einstellungen für Systemdiagnosen zu importieren:

```
public static void ImportSystemDignosticSettings(project tiaProject)
{
    Siemens.Engineering.HW.Systemdiagnostics.Settings.SystemdiagnosticsSettingsDataProvider
    settingsProvider =
    tiaProject.GetService<Siemens.Engineering.HW.Systemdiagnostics.Settings.SystemdiagnosticsS
    ettingsDataProvider>();
    Siemens.Engineering.HW.Systemdiagnostics.Settings.SystemdiagnosticsSettingsExportImportRes
    ult importResult = settingsProvider.Import(new FileInfo(@"D:\Temp\SysDiagSettings.dat"));
    Siemens.Engineering.HW.Systemdiagnostics.Settings.SystemdiagnosticsSettingsExportImportRes
    ultState resultState = importResult.State;
}
```

5.4 Funktionen für den Zugriff auf das Teamcenter Gateway

5.4.1 Verbindung zum Teamcenter Gateway herstellen

Voraussetzung

- Die Anwendung TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness über das TcGateway eine Verbindung zu Teamcenter herstellen.

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
userName	System.String	Ja	Gibt den Benutzernamen in Teamcenter an.
userPassword	System.SecurityString	Ja	Gibt das Passwort des Benutzers in Teamcenter an.
serverURL	System.String	Ja	Gibt die voll qualifizierte URL des Teamcenter-Servers an.
teamcenterInstance	System.String	Ja	Gibt die Instanz des Teamcenter-Servers an.
userRole	System.String	Nein	Gibt die in Teamcenter festgelegte Benutzerrolle an.
userGroup	System.String	Nein	Gibt die Gruppe an, zu der der Benutzer in Teamcenter gehört.

Hinweis

Wird einer der oben aufgeführten obligatorischen Parameter nicht übertragen, wird eine Siemens.Engineering-Ausnahme des Typs TcGatewayException veranlasst.

Bei erfolgreich hergestellter Verbindung wird ein verschlüsseltes Objekt TcGatewayConnectionInfo zurückgegeben. Wird die Verbindung nicht erfolgreich hergestellt,

wird eine Engineering-Ausnahme des Typs `TcGatewayException` mit Angabe der Fehlerursache ausgelöst.

Datentyp	Beschreibung
<code>TcGatewayConnectionInfo</code>	Das Objekt <code>TcGatewayConnectionInfo</code> muss zwingend in allen <code>TcGateway Openness APIs</code> übergeben werden. Es wird in anderen <code>TcGateway Openness APIs</code> zur Validierung der Aufrufquelle und der aktiven Teamcenter-Sitzung verwendet. <code>TcGatewayConnectionInfo</code> stellt die folgenden Verbindungs-details bereit: • <code>SessionToken</code> (aktiver Sitzungstoken für die Verbindung) • <code>Gruppe</code> (Teamcenter-Benutzergruppe) • <code>Rolle</code> (Benutzerrolle in Teamcenter)

Programmcode

```
//Connecting Teamcenter using TcGateway
public void function1()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal TIAInstance = ...;
        TeamcenterConnectionProvider tcGatewayProvider =
        TIAInstance.GetService<TeamcenterConnectionProvider>();
        String userName = ...; //User name
        SecureString password = ...; //Secure string. Client takes care of encrypting password and
        openness framework decrypts and passes the decrypted value to the action
        String userRole = ...; //Role
        String userGroup = ...; //Group
        String serverURL = ...; //Teamcenter server URL
        String teamcenterInstance=...; //Instance
        TcGatewayConnectionInfo connectionInfo = tcGatewayProvider.Connect(userName, password,
        userGroup, userRole, serverURL,teamcenterInstance);
        string group = connectionInfo.Group;
        string role = connectionInfo.Role;
    }
}
```

5.4.2 Verbindung zum Teamcenter Gateway trennen

Voraussetzung

- Die Anwendung TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können mit TIA Portal Openness die Verbindung zwischen dem Teamcenter-Server und dem TIA Portal über das TcGateway trennen.

Parameter

Parameter	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	TcGatewayConnection-Info	Ja	Gibt während des Aufrufs der API Connect / ConnectSSO das Objekt TcGatewayConnection-Info zurück.

Programmcode

```
public void function2()
{
    //Program code: Disconnect Teamcenter using TcGateway
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal TIAInstance = ...;
        TcGatewayConnectionProvider tcGatewayProvider =
        TIAInstance.GetService<TeamcenterConnectionProvider>();
        String userName = ...; //User name
        SecureString password = ...; //Secure string. Client takes care of encrypting password and
        openness framework decrypts and passes the decrypted value to the action
        String userRole = ...; //Role
        String userGroup = ...; //Group
        String serverURL = ...; //Teamcenter server URL
        String teamcenterInstance = ...; //Instance
        TcGatewayConnectionInfo tcgConnectionInfo = tcGatewayProvider.Connect(userName,
        password, serverURL, teamcenterInstance, userRole, userGroup);
        tcGatewayProvider.Disconnect(tcgConnectionInfo);
    }
}
```

Siehe auch

Verbindung zum Teamcenter Gateway herstellen (Seite 164)

5.4.3 Verbindung mit ConnectSSO herstellen

Voraussetzung

- Die Anwendung TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können mit TIA Portal Openness `ConnectSSO()` aufrufen, wenn eine aktive SSO-Sitzung mit den nachfolgend aufgeführten Pflichtparametern verfügbar ist.

Parameter	Datentyp	Obligatorisch	Beschreibung
hostURL	System.String	Ja	Gibt den voll qualifizierten Teamcenter-Serverhost mit Portnummer an.
instance	System.String	Ja	Gibt die Teamcenter-Serverinstanz an.
loginURL	System.String	Ja	Legt die URL für den im Rahmen der Teamcenter Security Services konfigurierten Login-Dienst fest.
applicationID	System.String	Ja	Gibt die in der Registry der Teamcenter-Anwendung über den Identity-Dienst konfigurierte ID der Anwendung an.

Bei erfolgreich hergestellter Verbindung wird ein verschlüsseltes Objekt `TcGatewayConnectionInfo` zurückgegeben. Wird die Verbindung nicht erfolgreich hergestellt, wird eine entsprechende Ausnahme gemeldet.

Hinweis

Ist keine aktive SSO-Sitzung verfügbar, werden Sie aufgefordert, Ihre Anmeldedaten für Teamcenter im Browser einzugeben, um eine SSO-Sitzung zu erstellen. Die API unterstützt das Anmelden mit Secure Card über PKI.

Programmcode

```
//Program code: ConnectSSO using TcGatewayConnectionInfo
public void function3()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal TIAInstance = ...;
        TeamcenterConnectionProvider tcGatewayProvider =
        TIAInstance.GetService<TeamcenterConnectionProvider>();
        String hostUrl = ...;//Fully qualified Teamcenter server host with port number
        String teamcenterInstance=...;//Instance
        String loginURL = ...;//Teamcenter SSO login Service url
        String applicationId = ...;//Teamcenter SSO application
        TcGatewayConnectionInfo connectionInfo =
        tcGatewayProvider.ConnectSSO(hostUrl,teamcenterInstance,loginURL,applicationId);
        string group = connectionInfo.Group;
        string role = connectionInfo.Role;
    }
}
```

5.4.4 Projekt/globale Bibliothek in Teamcenter speichern

Voraussetzung

- Die Anwendung TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness ein Projekt oder eine globale Bibliothek über den Dienst TcGatewayWorkflowProvider speichern. Dieser Dienst steht bei der Zusammensetzung des Projekts/der globalen Bibliothek zur Verfügung.

5.4 Funktionen für den Zugriff auf das Teamcenter Gateway

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurück. tcGatewayConnectionInfo wurde übergeben und stimmt mit der tcGatewayConnectionInfo der aktiven Verbindung überein.
localCacheOption	LocalCacheOption(Enum)	Ja	TcGateway LocalCacheOption überschreibt das TIA-Projekt/die globale Bibliothek im Teamcenter-Cache.

Data Type	Beschreibung
Siemens.Engineering.TeamcenterGateway.ItemInfo	Wenn das Projekt / die globale Bibliothek erfolgreich in Teamcenter gespeichert ist, gibt die API das folgende Objekt ItemInfo zurück: • ItemId [System.String] • RevisionId [System.String] • ItemName [System.String] • ItemType [Siemens.Engineering.TeamcenterGateway.ItemType(Enum)]

Programmcode

```
//Program code: Saving Project/global library in Teamcenter using Save()
public void function4()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal tiaPortal = ...;
        var tcGatewayConnectionProvider= tiaPortal.GetService<TeamcenterConnectionProvider>();
        var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
        try
        {
            String userName = "name";//User name
            SecureString password = ...;//Secure string. Client takes care of encrypting password and
            openness framework decrypts passes the decrypted value to the action
            String userRole = ...;//Role
            String userGroup = ...;//Group
            String serverURL = ...;//Teamcenter server URL
            String teamcenterInstance = ...;//Instance
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL,teamcenterInstance);
            tcGatewayLockProvider.CheckoutDataset(connectionInfo ,"000495","A",DatasetType.T4TiaProject
            Dataset,"Project30");
            ProjectComposition projectComposition = tiaPortal.Projects;
            String projectPath = ...; // Project path for the checkout dataset in above step or download
            project using TcGateway Load API Project
            Project tcgProject = projectComposition.Open(new FileInfo(projectPath));
            tcgProject.DeviceGroups.Create("Group_1");
            tcgProject.Save(); //This is used to save project locally in TC cache. (optional)
            TcGatewayWorkflowProvider tcgWorkflowProvider =
            tcgProject.GetService<TcGatewayWorkflowProvider>();
            ItemInfo response = tcgWorkflowProvider.Save(connectionInfo , LocalCacheOption.Overwrite);
            //OR
            ItemInfo response = tcgWorkflowProvider.Save(connectionInfo ,
            LocalCacheOption.DoNotOverwrite);
            tcGatewayLockProvider.CheckinDataset(connectionInfo ,"000495","A",DatasetType.T4TiaProject
            Dataset,"Project30");
        }
        catch (TcGatewayException)
        {
            // other operations
        }
    }
}
```

5.4.5 Projekt/globale Bibliothek mit Proxy-Objekt speichern

Voraussetzung

- Die Anwendung TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness das Projekt / die globale Bibliothek archivieren und das archivierte Objekt zusammen mit Proxy-Objekten (z. B. PLC, Programmbausteine, UDTs, andere unterstützte Typen) in den Teamcenter-Datensatz speichern.

Sobald das Projekt / die globale Bibliothek erfolgreich im Teamcenter gespeichert ist, gibt die API Objekt ItemInfo mit Details zu ItemId, RevisionId, ItemName und ItemType zurück.

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurück. tcGatewayConnectionInfo muss mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen
localCacheOption	LocalCacheOption(Enum)	Ja	TcGateway LocalCacheOption überschreibt TIA Projekt/globale Bibliothek im TC-Cache.

Programmcode: Projekt/globale Bibliothek in Teamcenter speichern mit SaveWithProxyObject()

```
//Program code: Saving project/ global library in Teamcenter using SaveWithProxyObject()
public void function5()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal tiaPortal = ...;
        var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
        var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
        try
        {
            String userName = string.Empty; //User name
            SecureString password = new SecureString(); //Secure string. Client takes care of encrypting
            password and openness framework decrypts and passes the decrypted value to the action
            String userRole = ...; //Role
            String userGroup = ...; //Group
            String serverURL = string.Empty; //Teamcenter server URL
            String teamcenterInstance = string.Empty; //Instance
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL, teamcenterInstance);
            tcGatewayLockProvider.CheckoutDataset(connectionInfo, "000495", "A",
            DatasetType.T4TiaProjectDataset, "Project30");
            ProjectComposition projectComposition = tiaPortal.Projects;
            String projectPath = string.Empty; // Project path for the checkout dataset in above step or
            download project using TcGateway Load API
            Project tcgProject = projectComposition.Open(new FileInfo(projectPath));
            TcGatewayWorkflowProvider tcgWorkflowProvider =
            tcgProject.GetService<TcGatewayWorkflowProvider>();
            ItemInfo response = tcgWorkflowProvider.SaveWithProxyObject(connectionInfo,
            LocalCacheOption.Overwrite);
            //OR
            ItemInfo response = tcgWorkflowProvider.SaveWithProxyObject(connectionInfo,
            LocalCacheOption.DoNotOverwrite);
            tcGatewayLockProvider.CheckinDataset(connectionInfo, "000495", "A",
            DatasetType.T4TiaProjectDataset, "Project30");
        }
        catch (TcGatewayException)
        {
            // other operations
        }
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

5.4.6 Projekt/globale Bibliothek mit neuem Element speichern

Voraussetzung

- Verbindung zum TIA Portal herstellen
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Gibt tcGatewayConnectionInfo für die erfolgreich hergestellte Verbindung zu Teamcenter zurück. tcGatewayConnectionInfo wurde übergeben und muss mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen.
itemId	System.String	Ja	Gibt die ID an, unter der das Projekt gespeichert werden muss.
revisionId	System.String	Ja	Gibt die Versions-ID an, unter der das Projekt gespeichert werden muss.
localCacheOption	Siemens.Engineering.TeamcenterGateway.LocalCacheOption(Enum)	Ja	Gibt die zu überschreibenden Projekteinhalte des lokalen Caches an, wenn die lokale Cache- oder Datensatz-Version des Projekts nicht mit dem Serverprojekt übereinstimmt.

Programmcode

```
//Program code: Saving project/global library in Teamcenter using SaveToItem( )
public void function6()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal tiaPortal = ...;
        var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
        var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
        try
        {
            String userName = string.Empty; //User name
            SecureString password = new SecureString(); //Secure string. Client takes care of encrypting
            password and openness framework decrypts and passes the decrypted value to the action
            String userRole = ...; //Role
            String userGroup = ...; //Group
            String serverURL = string.Empty; //Teamcenter server URL
            String teamcenterInstance = string.Empty; //Instance
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL, teamcenterInstance);
            tcGatewayLockProvider.CheckoutDataset(connectionInfo, "000495", "A",
            DatasetType.T4TiaProjectDataset, "Project30");
            ProjectComposition projectComposition = tiaPortal.Projects;
            String projectPath = string.Empty; // Project path for the checkout dataset in above step or
            download project using TcGateway Download API
            Project tcgProject = projectComposition.Open(new FileInfo(projectPath));
            TcGatewayWorkflowProvider tcgWorkflowProvider =
            tcgProject.GetService<TcGatewayWorkflowProvider>();
            ItemInfo response = tcgWorkflowProvider.SaveToItem(connectionInfo, "000495", "A",
            LocalCacheOption.Overwrite);
            tcGatewayLockProvider.CheckinDataset(connectionInfo, "000495", "A",
            DatasetType.T4TiaProjectDataset, "Project30");
        }
        catch (TcGatewayException)
        {
            // other operations
        }
    }
}
```

5.4.7 Auf benutzerdefinierte Attribute in Teamcenter zugreifen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Mit TIA Portal Openness können Sie die benutzerdefinierten Attributwerte im Teamcenter Gateway abrufen und einstellen.

Parameter

Das itemType in Teamcenter bezieht sich auf T4TiaLibrary, T4TiaProject und Subtypen im Teamcenter-Datenmodell, für die benutzerdefinierte Attribute in Teamcenter dargestellt werden.

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurück. tcGatewayConnectionInfo wurde übergeben und muss mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen.
itemType	System.String	Ja	Gibt den Teamcenter-itemType an.

Nach der erfolgreichen Ausführung gibt die API eine Sammlung benutzerdefinierter Attribute zurück, wenn diese in Teamcenter für den angegebenen itemType verfügbar sind.

Datentyp	Beschreibung
IList<Siemens.Engineering.TeamcenterGateway.TeamcenterProperty>	Gibt die Liste der benutzerdefinierten Eigenschaften in Teamcenter für das Objekt oder die Version an. - Name [System.String] - MappingLevel [System.int] - ListOfValueInfo [Siemens.Engineering.TeamcenterGateway.TcPropertyListOfValueInfo] - LowerLimit [System.String] - UpperLimit [System.String] - Values [System.Collections.Generic.IEnumerable<System.String>] - DataType [Siemens.Engineering.TeamcenterGateway.MappedCustomAttributeType(Enum)]

Die API Call SetValue setzt den Wert für benutzerdefinierte Attribute, die mit Hilfe der API GetTeamcenterCustomAttributes abgefragt werden, zusammen mit dem errorCallback-Delegate.

Parametername	Datentyp	Obligatorisch	Beschreibung
value	System.String	Ja	Gibt den Wert für das benutzerdefinierte Attribut an.
errorCallback	Siemens.Engineering.TeamcenterGateway.ErrorCallback	Ja	Delegate für die Fehlerbehandlung bei Aufrufen/Aktionen der API.

Programmcode

```

public void function7()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
        {
            var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
            var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
            try
            {
                String userName = string.Empty; //User name
                SecureString password = new SecureString(); //Secure string. Client takes care of encrypting
                password and openness framework decrypts and passes the decrypted value to the action
                String userRole = ...; //Role
                String userGroup = ...; //Group
                String serverURL = string.Empty; //Teamcenter server URL
                String teamcenterInstance = string.Empty; //Instance
                ItemDetailsDelegate itemDetailsDelegate = ItemDetailsForCreation;
                TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
                password, userGroup, userRole, serverURL, teamcenterInstance);
                var project = tiaPortal.Projects.Open(new FileInfo("D:\\UserData\\Documents\\Automation\\
                \\Project30\\Project30.ap18"));
                var tcGatewayWorkflowProvider = project.GetService<TcGatewayWorkflowProvider>();
                //OR
                var library = tiaPortal.GlobalLibraries.Open(new
                FileInfo(@"~\Documents\Automation\Library7\Library7.all8"), OpenMode.ReadWrite);
                var tcGatewayWorkflowProvider = library.GetService<TcGatewayWorkflowProvider>();
                //Get custom attributes.
                var attributes = tcGatewayWorkflowProvider.GetTeamcenterCustomAttributes(connectionInfo,
                "T4TiaProject");
                ErrorCallback errorCallback = PropertyValueSettingErrorCallback;
                foreach (var arr in attributes)
                {
                    if (arr.ListOfValueInfo != null)
                    {
                        foreach (var l in arr.ListOfValueInfo.Values)
                        {
                            Console.WriteLine(l);
                        }
                    }
                }
                switch (arr.DataType)
                {
                    {
                        case MappedCustomAttributeType.String:
                            arr.SetValue("23232323232", errorCallback);
                            break;
                        case MappedCustomAttributeType.Char:
                            arr.SetValue("d", errorCallback);
                            break;
                        case MappedCustomAttributeType.Integer:
                            arr.SetValue("232", errorCallback);
                            break;
                        case MappedCustomAttributeType.arr.SetValue(23.4f.ToString(), errorCallback);
                            break;
                    }
                }
            }
        }
    }
}

```

```
}  
catch (TcGatewayException)  
{  
    // other operations  
}  
}  
}  
}
```

5.4.8 Projekt in Teamcenter als neues Element speichern

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Hotspot-TextVerbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können mit TIA Portal Openness ein TIA Portal-Projekt als neues Element in Teamcenter speichern. Die API gibt das Objekt itemInfo mit ItemId, RevisionId, ItemType, ItemName zurück, nachdem das Element/die Version erfolgreich angelegt wurde.

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurück. tcGatewayConnectionInfo muss mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen.
itemDetailsDelegate	delegate	Ja	Dieses Delegate wird vom Aufrufer der API mit Informationen über das in Teamcenter anzulegende Element festgelegt: • ItemId [System.String] • RevisionId [System.String] • ItemName [System.String](Required) • Comments [System.String] • TeamcenterFolder [System.String] • TeamcenterProject [System.String[]] • TeamcenterItemType [System.String](Required)
customAttributes	IEnumerable<Siemens.Engineering.TeamcenterGateway.TeamcenterProperty>	Nein	Gibt die benutzerdefinierten Attribute zu Element/Version an, die zusammen mit Projekt / globaler Bibliothek gespeichert werden.

Programmcode

```
private static void ItemDetailsForCreation(ItemDetails itemDetails)
{
    itemDetails.ItemId = "...";
    itemDetails.ItemName = "Proj_Object";
    itemDetails.RevisionId = "...";
    itemDetails.TeamcenterItemType = "T4TiaProject";
    itemDetails.Comment = "Prooject Comments.";
    itemDetails.TeamcenterFolder = "Home\\TIA_Project\\ConveyourBelt";
    itemDetails.TeamcenterProject = new string[] { "Project_TIA_1", "Project_TIA_2" };
}
```

5.4 Funktionen für den Zugriff auf das Teamcenter Gateway

```
private static void PropertyValueSettingErrorCallback(string errorMessage)
{
    Console.WriteLine(errorMessage);
}
```



```
//Program code: Saving project to Teamcenter as new item
private void functionSavingProjectToTeamcenterAsNewItem()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal tiaPortal = ...;
        var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
        var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
        try
        {
            String userName = string.Empty; //User name
            SecureString password = new SecureString(); //Secure string. Client takes care of encrypting
            password and openness framework decrypts and passes the decrypted value to the action
            String userRole = ...; //Role
            String userGroup = string.Empty; //Group
            String serverURL = string.Empty; //Teamcenter server URL
            String teamcenterInstance = string.Empty; //Instance
            ItemDetailsDelegate itemDetailsDelegate = ItemDetailsForCreation;
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL, teamcenterInstance);
            var project = tiaPortal.Projects.Open(new FileInfo("D:\\UserData\\Documents\\Automation\\
            \\Project30\\Project30.ap18"));
            var tcGatewayWorkflowProvider = project.GetService<TcGatewayWorkflowProvider>();
            //OR
            var library = tiaPortal.GlobalLibraries.Open(new
            FileInfo(@"~\Documents\Automation\Library7\Library7.all8"), OpenMode.ReadWrite);
            var tcGatewayWorkflowProvider = library.GetService<TcGatewayWorkflowProvider>();
            //Save item in Teamcenter with custom attributes.
            var attributes = tcGatewayWorkflowProvider.GetTeamcenterCustomAttributes(connectionInfo,
            "T4TiaProject");
            ErrorCallback errorCallback = PropertyValueSettingErrorCallback;
            foreach (var arr in attributes)
            {
                if (arr.ListOfValueInfo != null)
                {
                    //Err:
                    //Invalid arr.ListOfValueInfo element
                    foreach (var l in arr.ListOfValueInfo?.Values)
                    {
                        Console.WriteLine(l);
                    }
                }
                switch (arr.DataType)
                {
                    {
                        case MappedCustomAttributeType.String:
                            arr.SetValue("23232323232", errorCallback);
                            break;
                        case MappedCustomAttributeType.Char:
                            arr.SetValue("d", errorCallback);
                            break;
                        case MappedCustomAttributeType.Integer:
                            arr.SetValue("232", errorCallback);
                            break;
                        case MappedCustomAttributeType.Float:
                            arr.SetValue(23.4f.ToString(), errorCallback);
                    }
                }
            }
        }
    }
}
```

5.4 Funktionen für den Zugriff auf das Teamcenter Gateway

```
break;
}
var createdItem = tcGatewayWorkflowProvider.SaveAsNewItem(connectionInfo,
itemDetailsDelegate, attributes);
//OR
var createdItem = tcGatewayWorkflowProvider.SaveAsNewItem(connectionInfo,
itemDetailsDelegate, null);//< --optional
var itemId = createdItem.ItemId;
var revisionId = createdItem.RevisionId;
var itemName = createdItem.ItemName;
var type = createdItem.ItemType;
//Checkout created dataset to edit in TIA portal
tcGatewayLockProvider.CheckoutDataset(connectionInfo, itemId, revisionId,
DatasetType.T4TiaProjectDataset, itemName);
//Check in dataset after changes to project are completed
tcGatewayLockProvider.CheckinDataset(connectionInfo, itemId, revisionId,
DatasetType.T4TiaProjectDataset, itemName);
}
}
catch (TcGatewayException)
{
// other operations
}
}
```

5.4.9 Projekt als neues Element mit Proxy-Objekt speichern

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness das TIA Portal-Projekt und die globale Bibliothek zusammen mit einem Proxy-Objekt in Teamcenter speichern. Die API gibt Details zum Objekt itemInfo – beispielsweise ItemId, RevisionId, ItemType oder ItemName – zurück, nachdem das Element/die Version erfolgreich angelegt wurde.

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurück. tcGatewayConnectionInfo wurde übergeben und muss mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen.
itemDetailsDelegate	delegate	Ja	Dieses Delegate wird vom Aufrufer der API mit Informationen über das in Teamcenter anzulegende Element festgelegt. • ItemId [System.String] • RevisionId [System.String] • ItemName [System.String](Required) • Comments [System.String] • TeamcenterFolder [System.String] • TeamcenterProject [System.String[]] • TeamcenterItemType [System.String](Required)
customAttributes	IEnumerable<Siemens.Engineering.TeamcenterGateway.TeamcenterProperty>	Nein	Gibt die benutzerdefinierten Attribute zu Element/Version an, die zusammen mit Projekt / globaler Bibliothek gespeichert werden.

Programmcode

```
private static void ItemDetailsForCreation(ItemDetails itemDetails)
{
    itemDetails.ItemId = "...";
    itemDetails.ItemName = "Proj_Object";
    itemDetails.RevisionId = "...";
    itemDetails.TeamcenterItemType = "T4TiaProject";
    itemDetails.Comment = "Project Comments.";
    itemDetails.TeamcenterFolder = "Home\\TIA_Project\\ConveyourBelt";
    itemDetails.TeamcenterProject = new string[] { "Project_TIA_1", "Project_TIA_2" };
}

private static void PropertyValueSettingErrorCallback(string errorMessage)
{
    Console.WriteLine(errorMessage);
}
```


5.4 Funktionen für den Zugriff auf das Teamcenter Gateway

```
//Program code: Saving project/library in Teamcenter as new item with proxy object
private void function SavingProjectAsNewItemWithProxyObject()
{
using Siemens.Engineering.TeamcenterGateway;
{
TiaPortal tiaPortal = ...;
var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
try
{
String userName = string.Empty; //User name
SecureString password = new SecureString(); //Secure string. Client takes care of encrypting
password and openness framework decrypts and passes the decrypted value to the action
String userRole = string.Empty; //Role
String userGroup = string.Empty; //Group
String serverURL = string.Empty; //Teamcenter server URL
String teamcenterInstance = string.Empty; //Instance
ItemDetailsDelegate itemDetailsDelegate = ItemDetailsForCreation;
TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
password, userGroup, userRole, serverURL, teamcenterInstance);
var project = tiaPortal.Projects.Open(new FileInfo("D:\\UserData\\Documents\\Automation\\
\\Project30\\Project30.apl8"));
var tcGatewayWorkflowProvider = project.GetService<TcGatewayWorkflowProvider>();
//OR
var library = tiaPortal.GlobalLibraries.Open(new
FileInfo(@"~\Documents\Automation\Library7\Library7.all8"), OpenMode.ReadWrite);
var tcGatewayWorkflowProvider = library.GetService<TcGatewayWorkflowProvider>();
//Save item in Teamcenter with custom attributes.
var attributes = tcGatewayWorkflowProvider.GetTeamcenterCustomAttributes(connectionInfo,
"T4TiaProject");
ErrorCallback errorCallback = PropertyValueSettingErrorCallback;
foreach (var arr in attributes)
{
if (arr.ListOfValueInfo != null)
{
foreach (var l in arr.ListOfValueInfo?.Values)
{
Console.WriteLine(l);
}
}
switch (arr.DataType)
{
case MappedCustomAttributeType.String:
arr.SetValue("23232323232", errorCallback);
break;
case MappedCustomAttributeType.Char:
arr.SetValue("d", errorCallback);
break;
case MappedCustomAttributeType.Integer:
arr.SetValue("232", errorCallback);
break;
case MappedCustomAttributeType.Float:
arr.SetValue(23.4f.ToString(), errorCallback);
break;
}
}
}
```

```

var createdItem = tcGatewayWorkflowProvider.SaveAsNewItemWithProxyObject(connectionInfo,
itemDetailsDelegate, attributes);
//OR
var createdItem = tcGatewayWorkflowProvider.SaveAsNewItem(connectionInfo,
itemDetailsDelegate, null); //< --optional
var itemId = createdItem.ItemId;
var revisionId = createdItem.RevisionId;
var itemName = createdItem.ItemName;
var type = createdItem.ItemType;
//Checkout created dataset to edit in TIA portal
tcGatewayLockProvider.CheckoutDataset(connectionInfo, itemId, revisionId,
DatasetType.T4TiaProjectDataset, itemName);
//Check in dataset after changes to project are complete
tcGatewayLockProvider.CheckinDataset(connectionInfo, itemId, revisionId,
DatasetType.T4TiaProjectDataset, itemName);
}
}
catch (TcGatewayException)
{
// other operations
}
}
}

```

5.4.10 Projekt in Teamcenter als neue Version speichern

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness das TIA Portal-Projekt und die globale Bibliothek als neue Version in Teamcenter speichern. Die API gibt Objekt ItemInfo mit ItemId, RevisionId, ItemName und ItemType zurück, wenn die neue Version erfolgreich angelegt wurde.

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurück. tcGatewayConnectionInfo wurde übergeben und muss mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen.
revisionDetailsDelegate	RevisionDetailsDelegate	Nein	Dieses Delegate wird vom Aufrufer der API mit Informationen über die in Teamcenter anzulegende Version festgelegt.
customAttributes	IEnumerable<Siemens.Engineering.TeamcenterGateway.TeamcenterProperty>	Nein	Gibt die benutzerdefinierten Attribute zu Element/Version an, die zusammen mit Projekt / globaler Bibliothek gespeichert werden.

Programmcode

```
//Program code: Saving project/library in Teamcenter as new revision
public static void RevisionDetailsForCreation(RevisionDetails revisionDetails)
{
    revisionDetails.RevisionId = "B";
    revisionDetails.Comment = "Revised project";
}
```


5.4 Funktionen für den Zugriff auf das Teamcenter Gateway

```

private void SavingProjectToTeamcenterAsNewRevision()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal tiaPortal = ...;
        var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
        var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
        try
        {
            String userName = string.Empty; //User name
            SecureString password = new SecureString(); //Secure string. Client takes care of encrypting
            password and openness framework decrypts and passes the decrypted value to the action;
            String userRole = ...; //Role
            String userGroup = ...; //Group
            String serverURL = string.Empty; //Teamcenter server URL
            String teamcenterInstance = string.Empty; //InstanceRevisionDetailsDelegate
            revisionDetailsDelegate = RevisionDetailsForCreation;
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL, teamcenterInstance);
            var project = tiaPortal.Projects.Open(new FileInfo("D:\\UserData\\Documents\\Automation\\
            \\Project30\\Project30.apl8"));
            var tcGatewayWorkflowProvider = project.GetService<TcGatewayWorkflowProvider>();
            ItemDetailsDelegate itemDetailsDelegate = ItemDetailsForCreation;
            void ItemDetailsForCreation(ItemDetails itemDetails)
            {
                itemDetails.ItemId = "...";
                itemDetails.ItemName = "Project30";
                itemDetails.RevisionId = "...";
                itemDetails.TeamcenterItemType = "T4TiaProject";
                itemDetails.Comment = "Project Comments.";
            }
            var saveAsNewItemResult = tcGatewayWorkflowProvider.SaveAsNewItem(connectionInfo,
            itemDetailsDelegate, null);
            //Checkout created dataset to edit in TIA portal
            tcGatewayLockProvider.CheckoutDataset(connectionInfo, "000495",
            "A"), DatasetType.T4TiaProjectDataset, "Project30");
            var saveAsNewRevisionResult = tcGatewayWorkflowProvider.SaveAsNewRevision(connectionInfo,
            revisionDetailsDelegate, null);
            //Check in dataset after changes to project are complete
            tcGatewayLockProvider.CheckinDataset(connectionInfo, "000495", "A",
            DatasetType.T4TiaProjectDataset, "Project30");
            //Checkout revised project/library
            tcGatewayLockProvider.CheckoutDataset(connectionInfo, "000495", "B",
            DatasetType.T4TiaProjectDataset, "Project30");
            //Check in revised dataset after changes to project are complete
            tcGatewayLockProvider.CheckinDataset(connectionInfo, "000495", "B",
            DatasetType.T4TiaProjectDataset, "Project30");
        }
        catch (TcGatewayException)
        {
            // other operations
        }
    }
}

```

5.4.11 Projekt/globale Bibliothek aus Teamcenter laden

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness ein Projekt und eine Bibliothek aus Teamcenter laden und öffnen. Die API gibt den Startdateipfad zum Öffnen des Projekts oder der Bibliothek im TIA Portal zurück.

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurück. tcGatewayConnectionInfo muss mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen.
itemID	System.String	Ja	Gibt die ID für das TIA-Projekt / die Bibliothek an.
revisionID	System.String	Ja	Gibt die Versions-ID für das TIA-Projekt / die Bibliothek an.
itemType	Siemens.Engineering.TeamcenterGateway.ItemType(Enum)	Ja	ItemType für das Projekt/die Bibliothek in Teamcenter.
localCacheOption	Siemens.Engineering.TeamcenterGateway.LocalCacheOption(Enum)	Ja	Inhalt des TcCache wird überschrieben, wenn bereits eine Version des im Laden begriffenen Projekts / der globalen Bibliothek vorhanden ist. "True" bedeutet, der Inhalt wird überschrieben. Bei "False" wird eine Ausnahme erzeugt, wenn die Versionen zwischen TcCache und Teamcenter nicht übereinstimmen oder wenn Inhalte geändert wurden.

Hinweis

Wird einer der obligatorischen Parameter nicht übertragen, wird eine Siemens.Engineering-Ausnahme des Typs TcGatewayException veranlasst.

Programmcode

```
//Program code: Download project/library in Teamcenter
private void DownloadProjectLibrary()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal tiaPortal = ...;
        var tcGatewayConnectionProvider= tiaPortal.GetService<TeamcenterConnectionProvider>();
        var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
        String userName = ...;//User name
        SecureString password = ...;//Secure string. Client takes care of encrypting password and
        openness framework decrypts and passes the decrypted value to the action;
        String userRole = ...;//Role
        String userGroup = ...;//Group
        String serverURL = ...;//Teamcenter server URL
        String teamcenterInstance = ...;//Instance
        try
        {
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL, teamcenterInstance);
            tcGatewayLockProvider.CheckoutDataset(connectionInfo, "000495", "A", DatasetType.T4TiaProjec
            tDataset, "Project30");
            TcGatewaySearchAndDownloadProvider downloadFromTcProvider =
            tiaPortal.GetService<TcGatewaySearchAndDownloadProvider>();
            FileInfo TIAObjectStarterFilePath =
            downloadFromTcProvider.Download(connectionInfo, "000495", "A", ItemType.Project,
            LocalCacheOption.Overwrite);
        }
        catch (TcGatewayException)
        {
            // other operations
        }
        var project = tiaPortal.Projects.Open(TIAObjectStarterFilePath);
        //OR
        var library=tiaPortal.GlobalLibraries.Open(TIAObjectStarterFilePath, OpenMode.ReadWrite);
        //User can work on the Project.
        project.DeviceGroups.Create("Group_3");
        project.Save();
    }
}
```

5.4.12 Projekt/globale Bibliothek in Teamcenter suchen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness ein Projekt und die globale Bibliothek in Teamcenter suchen. Die API gibt eine Liste der TIA Portal-Objekte in Teamcenter mit dem Suchergebnis zurück, unter Angabe von Details wie ItemID und Liste der zugehörigen RevisionIDs.

Parameter

Parameter	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGateway-ConnectionInfo	Ja	Gibt nach erfolgreichem Verbindungsaufbau connection-Info von Teamcenter zurück.
type	Siemens.Engineering.TeamcenterGateway.Item-Type(Enum)	Ja	Gibt den ItemType für das Projekt / die Bibliothek in Teamcenter an.
TIAObjectName	System.String	Nein	Gibt den Namen des TIA-Projekts / der Bibliothek an. Stellt einen kompletten Namen oder Wildcard-Zeichen bereit. [In mindestens einem der Felder Item ID, ItemName, TIAObjectName muss ein Wert eingegeben werden].
itemID	System.String	Nein	Gibt die ID für das TIA-Projekt / die Bibliothek an. Stellt eine komplette ItemId oder Wildcard-Zeichen bereit. [In mindestens einem der Felder Item ID, ItemName, TIAObjectName muss ein Wert eingegeben werden]

5.4 Funktionen für den Zugriff auf das Teamcenter Gateway

Parameter	Datentyp	Obligatorisch	Beschreibung
itemName	System.String	Nein	Gibt den Namen des TIA-Projekts / der Bibliothek an. Eingabe eines kompletten Namens oder von Wildcard-Zeichen. [In mindestens einem der Felder Item ID, ItemName, TIAObjectName muss ein Wert eingegeben werden]
revisionID	System.String	Nein	Gibt die Versions-ID für das TIA-Projekt / die Bibliothek an. Eingabe einer kompletten Versions-ID oder von Wildcard-Zeichen.

Hinweis

Werden die obligatorischen Parameter für "Projekt / Bibliothek in Teamcenter suchen" nicht übertragen, wird eine Siemens.Engineering-Ausnahme des Typs TcGatewayException ausgelöst.

Programmcode

```
//Program code: Search project/library in Teamcenter
private void function9()
{
    using Siemens.Engineering.TeamcenterGateway;
    using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
    {
        var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
        String userName = ...;//User name
        SecureString password = ...;//Secure string. Client takes care of encrypting password and
        openness framework decrypts and passes the decrypted value to the action
        String userRole = ...;//Role
        String userGroup = ...;//Group
        String serverURL = ...;//Teamcenter server URL
        String teamcenterInstance =...;//Instance
        try
        {
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL, teamcenterInstance);
            TcGatewaySearchAndDownloadProvider searchAndLoadProvider =
            tiaPortal.GetService<TcGatewaySearchAndDownloadProvider>();
            var items = searchAndLoadProvider.Search(connectionInfo, ItemType.Project, "P*", "", "",
            "");
        }
        catch (TcGatewayException)
        {
            // other operations
            foreach (var item in items)
            {
                var itemId = item.ItemId;
                var revisionIdList = item.RevisionId;
                foreach (var revision in revisionIdList)
                {
                    Console.WriteLine(itemId + "-" + revision);
                }
            }
        }
    }
}
```

5.4.13 Datensatz für Projekt/globale Bibliothek in Teamcenter auschecken

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Projekt öffnen (Seite 131)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 84)

Verwendung

Sie können mit TIA Portal Openness einen Datensatz für ein TIA Portal-Projekt bzw. die globale Bibliothek im Teamcenter auschecken.

Parameter

Parameter	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurück. tcGatewayConnectionInfo muss übergeben werden und mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen.
itemID	System.String	Ja	Gibt die ID für das TIA-Projekt / die Bibliothek in Teamcenter an.
revisionID	System.String	Ja	Gibt die Versions-ID für das TIA-Projekt / die Bibliothek in Teamcenter an, mit dem/der der Datensatz verbunden ist.
datasetType	Siemens.Engineering.TeamcenterGateway.DatasetType	Ja	Gibt den Typ des Teamcenter-Datensatzes für ein(e) Projekt / Bibliothek an, der ausgecheckt werden soll. Beispiel: T4TiaProjectDataset / T4TiaLibraryDataset.
datasetName	System.String	Ja	Gibt den Namen des Teamcenter-Datensatzes für ein(e) Projekt / Bibliothek an, der ausgecheckt werden soll.

Hinweis

Werden die obligatorischen Parameter für die Aktion "Datensatz für Projekt / Bibliothek in Teamcenter auschecken" nicht übertragen, wird eine Siemens.Engineering-Ausnahme des Typs TcGatewayException ausgelöst.

Programmcode: Datensatz in Teamcenter auschecken

```
//Program code: Checkout project/global library in Teamcenter dataset
private void function10()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal tiaPortal = ...;
        var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
        var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
        try
        {
            String userName = ...; //User name
            SecureString password = ...; //Secure string. Client takes care of encrypting password and
            openness framework decrypts and passes the decrypted value to the action
            String userRole = ...; //Role
            String userGroup = ...; //Group
            String serverURL = ...; //Teamcenter server URL
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL, teamcenterInstance);
            tcGatewayLockProvider.CheckoutDataset(connectionInfo, "000495", "A",
            DatasetType.T4TiaProjectDataset, "Project30");
        }
        catch (TcGatewayException)
        {
            //User can perform saving of Changes of TIA Project to the checked out dataset or any other
            operations
        }
    }
}
```

5.4.14 Datensatz für Projekt/globale Bibliothek in Teamcenter einchecken**Voraussetzung**

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness einen Datensatz für ein TIA Portal-Projekt bzw. die globale Bibliothek im Teamcenter einchecken.

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Während des Verbindungsaufbaus zu Teamcenter tcGatewayConnectionInfo zurückgeben. tcGatewayConnectionInfo muss übergeben werden und mit der tcGatewayConnectionInfo der aktiven Verbindung übereinstimmen.
itemID	System.String	Ja	Gibt die ID für das TIA Portal-Projekt / die Bibliothek in Teamcenter an.
revisionID	System.String	Ja	Gibt die Versions-ID für das TIA Portal-Projekt / die Bibliothek in Teamcenter an, mit dem/der der Datensatz verbunden ist.
datasetType	Siemens.Engineering.TeamcenterGateway.DatasetType	Ja	Gibt den Typ des Teamcenter-Datensatzes für ein(e) Projekt / Bibliothek an, der eingecheckt werden soll. Beispiel: T4TiaProjectDataset / T4TiaLibraryDataset.
datasetName	System.String	Ja	Gibt den Namen des Teamcenter-Datensatzes für ein(e) Projekt / Bibliothek an, der eingecheckt werden soll.

Hinweis

Wird einer der oben aufgeführten obligatorischen Parameter für die Aktion "Datensatz für Projekt / Bibliothek in Teamcenter einchecken" nicht übertragen, wird eine Siemens.Engineering-Ausnahme des Typs TcGatewayException ausgelöst.

Programmcode

```
//Program code: Checkin project/global library in Teamcenter dataset
private void function11()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        using (TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface))
        {
            var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
            var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
            try
            {
                String userName = ...; //User name
                SecureString password = ...; //Secure string. Client takes care of encrypting password and
                openness framework decrypts and passes the decrypted value to the action
                String userRole = ...; //Role
                String userGroup = ...; //Group
                String serverURL = ...; //Teamcenter server URL
                String teamcenterInstance = ...; //Instance
                TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
                password, userGroup, userRole, serverURL, teamcenterInstance);
                tcGatewayLockProvider.CheckoutDataset(connectionInfo, "000495", "A",
                DatasetType.T4TiaProjectDataset, "Project30"); //User can perform saving of Changes of TIA
                Project to the checkedout dataset using TcGateway Save APIs.
                tcGatewayLockProvider.CheckinDataset(connectionInfo, "000495", "A",
                DatasetType.T4TiaProjectDataset, "Project30"); //Checkin to release the lock on dataset
            }
            catch (TcGatewayException)
            {
                // other operations
            }
        }
    }
}
```

5.4.15 Checkout für Datensatz in Teamcenter abbuchen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness das Auschecken eines Datensatzes für ein TIA Portal-Projekt und die globale Bibliothek in Teamcenter abbuchen. Sie können ferner `CancelCheckoutDataset()` verwenden, um die Sperre eines Datensatzes in Teamcenter aufzulösen.

Parameter

Parametername	Datentyp	Obligatorisch	Beschreibung
tcGatewayConnectionInfo	Siemens.Engineering.TeamcenterGateway.TcGatewayConnectionInfo	Ja	Gibt während des Verbindungsaufbaus zu Teamcenter <code>tcGatewayConnectionInfo</code> zurück. <code>tcGatewayConnectionInfo</code> muss übergeben werden und mit der <code>tcGatewayConnectionInfo</code> der aktiven Verbindung übereinstimmen.
itemID	System.String	Ja	Gibt die ID für das TIA-Projekt / die Bibliothek in Teamcenter an.
revisionID	System.String	Ja	Gibt die Versions-ID für das TIA-Projekt / die Bibliothek in Teamcenter an, mit dem/der der Datensatz verbunden ist.
datasetType	Siemens.Engineering.TeamcenterGateway.DatasetType	Ja	Gibt den Typ des Teamcenter-Datensatzes für ein(e) Projekt / Bibliothek an, der ausgecheckt werden soll. Beispiel: <code>T4TiaProjectDataset</code> / <code>T4TiaLibraryDataset</code> .
datasetName	System.String	Ja	Gibt den Namen des Teamcenter-Datensatzes für ein(e) Projekt / Bibliothek an, der ausgecheckt werden soll.

Hinweis

Wird einer der oben aufgeführten obligatorischen Parameter für den Abbruch der Aktion "Datensatz für Projekt / Bibliothek in Teamcenter auschecken" nicht übertragen, wird eine `Siemens.Engineering-Ausnahme` des Typs `TcGatewayException` ausgelöst.

Programmcode

```
//Program code: Cancel checkout dataset in Teamcenter
private void function12()
{
    using Siemens.Engineering.TeamcenterGateway;
    {
        TiaPortal tiaPortal = ...;
        var tcGatewayConnectionProvider = tiaPortal.GetService<TeamcenterConnectionProvider>();
        var tcGatewayLockProvider = tiaPortal.GetService<TcGatewayLockProvider>();
        try
        {
            String userName = ...;//User name
            SecureString password = ...;//Secure string. Client takes care of encrypting password and
            openness framework decrypts and passes the decrypted value to the action
            String userRole = ...;//Role
            String userGroup = ...;//Group
            String serverURL = ...;//Teamcenter server URL
            String teamcenterInstance = ...;//Instance
            TcGatewayConnectionInfo connectionInfo = tcGatewayConnectionProvider.Connect(userName,
            password, userGroup, userRole, serverURL, teamcenterInstance);
            //User can perform saving of Changes of TIA Project to the checked out dataset using
            TcGateway Save APIs.
            tcGatewayLockProvider.CheckoutDataset(connectionInfo, "000495", "A",
            DatasetType.T4TiaProjectDataset, "Project30");
            //Cancel checkout on dataset to release the lock on dataset and undo the changes done on
            dataset
            tcGatewayLockProvider.CancelCheckoutDataset(connectionInfo, "000495", "A",
            DatasetType.T4TiaProjectDataset, "Project30");
        }
        catch (TcGatewayException)
        {
            // other operations
        }
    }
}
```

5.5 Funktionen für Verbindungen

5.5.1 Konfigurierbare Attribute einer Port-zu-Port-Verbindung

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Projekt öffnen (Seite 131)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Die Attribute einer Portverschaltung befinden sich im Port-Geräteelement. Der Lese- und Schreibzugriff von Attributen über TIA Portal Openness ist mit dem in der UI identisch.

Einstellungen der Portschnittstelle

Die folgenden Attribute werden für die Einstellungen der Portschnittstelle bereitgestellt:

Attributname	Datentyp	Schreibbar	Zugriff	Beschreibung
MediumAttachmentType	MediumAttachment-Type	r/o	Dynamisches Attribut	-
CableName	CableName	r/w	Dynamisches Attribut	-
AlternativePartnerPorts	Bool	r/w	Dynamisches Attribut	Nur verfügbar, wenn die Funktionalität des Werkzeugwechslers unterstützt wird, z. B. bei der CPU1516.
SignalDelaySelection	SignalDelaySelection	r/w	Dynamisches Attribut	-
CableLength	CableLength	r/w	Dynamisches Attribut	-
SignalDelayTime	Double	r/w	Dynamisches Attribut	-

Die folgenden ENUM-Werte werden für das Attribut `MediumAttachmentType` bereitgestellt:

Wert	Beschreibung
<code>MediumAttachmentType.None</code>	Anbindungstyp kann nicht ermittelt werden.
<code>MediumAttachmentType.Copper</code>	Anbindungstyp ist Kupfer.
<code>MediumAttachmentType.FibreOptic</code>	Anbindungstyp ist Glasfaser.

Die folgenden ENUM-Werte werden für das Attribut `CableName` bereitgestellt:

Wert	Beschreibung
<code>CableName.None</code>	Kein Kabelname angegeben
<code>CableName.FO_Standard_Cable_9</code>	FO-Standardkabel GP (9 µm)
<code>CableName.Flexible_FO_Cable_9</code>	Flexibles FO-Kabel (9 µm)
<code>CableName.FO_Standard_Cable_GP_50</code>	FO-Standardkabel GP (50 µm)
<code>CableName.FO_Trailing_Cable_GP</code>	FO-Schleppkabel / GP
<code>CableName.FO_Ground_Cable</code>	FO-Erdungskabel
<code>CableName.FO_Standard_Cable_62_5</code>	FO-Standardkabel (62,5 µm)
<code>CableName.Flexible_FO_Cable_62_5</code>	Flexibles FO-Kabel (62,5 µm)
<code>CableName.POF_Standard_Cable_GP</code>	POF-Standardkabel GP
<code>CableName.POF_Trailing_Cable</code>	POF-Schleppkabel
<code>CableName.PCF_Standard_Cable_GP</code>	POF-Standardkabel GP
<code>CableName.PCF_Trailing_Cable_GP</code>	PCF-Schleppkabel / GP
<code>CableName.GI_POF_Standard_Cable</code>	GI-POF-Standardkabel

Wert	Beschreibung
CableName.GI_POF_Trailing_Cable	GI-POF-Schleppkabel
CableName.GI_PCF_Standard_Cable	GI-PCF-Standardkabel
CableName.GI_PCF_Trailing_Cable	GI-PCF-Schleppkabel

Die folgenden ENUM-Werte werden für das Attribut `SignalDelaySelection` bereitgestellt:

Wert	Beschreibung
SignalDelaySelection.None	-
SignalDelaySelection.CableLength	CableLength dient zur Festlegung der Signalverzögerung.
SignalDelaySelection.SignalDelayTime	CableDelayTime dient zur Festlegung der Signalverzögerung.

Die folgenden ENUM-Werte werden für das Attribut `CableLength` bereitgestellt:

Wert	Beschreibung
CableLength.None	Kabellänge nicht angegeben
CableLength.Length20m	Kabellänge ist 20 m.
CableLength.Length50m	Kabellänge ist 50 m.
CableLength.Length100m	Kabellänge ist 100 m.
CableLength.Length1000m	Kabellänge ist 1000 m.
CableLength.Length3000m	Kabellänge ist 3000 m.

Portoptionen

Die folgenden Attribute werden für Portoptionen bereitgestellt:

Attributname	Datentyp	Schreibbar	Zugriff
PortActivation	Bool	r/w	Dynamisches Attribut
TransmissionRateAndDuplex	TransmissionRateAndDuplex	r/w	Dynamisches Attribut
PortMonitoring	Bool	r/w	Dynamisches Attribut
TransmissionRateAutoNegotiation	Bool	r/w	Dynamisches Attribut
EndOfDetectionOfAccessibleDevices	Bool	r/w	Dynamisches Attribut
EndOfTopologyDiscovery	Bool	r/w	Dynamisches Attribut
EndOfSyncDomain	Bool	r/w	Dynamisches Attribut

Die folgenden ENUM-Werte werden für das Attribut `TransmissionRateAndDuplex` bereitgestellt:

Wert	Beschreibung
TransmissionRateAndDuplex.None	-
TransmissionRateAndDuplex.Automatic	Automatisch
TransmissionRateAndDuplex.AUI10Mbps	10 Mbps AUI

5.5 Funktionen für Verbindungen

Wert	Beschreibung
TransmissionRateAndDuplex.TP10MbpsHalfDuplex	TP 10 Mbps Halbduplex
TransmissionRateAndDuplex.TP10MbpsFullDuplex	TP 10 Mbps Vollduplex
TransmissionRateAndDuplex.AsyncFiber10MbpsHalfDuplex	Asynchrone Glasfaser 10 Mbit/s Halbduplexmodus
TransmissionRateAndDuplex.AsyncFiber10MbpsFullDuplex	Asynchrone Glasfaser 10 Mbit/s Vollduplexmodus
TransmissionRateAndDuplex.TP100MbpsHalfDuplex	TP 100 Mbps Halbduplex
TransmissionRateAndDuplex.TP100MbpsFullDuplex	TP 100 Mbps Vollduplex
TransmissionRateAndDuplex.FO100MbpsFullDuplex	FO 100 Mbps Vollduplex
TransmissionRateAndDuplex.X1000MbpsFullDuplex	X1000 Mbps Vollduplex
TransmissionRateAndDuplex.FO1000MbpsFullDuplexLD	FO 1000 Mbps Vollduplex LD
TransmissionRateAndDuplex.FO1000MbpsFullDuplex	FO 1000 Mbps Vollduplex
TransmissionRateAndDuplex.TP1000MbpsFullDuplex	TP 1000 Mbps Vollduplex
TransmissionRateAndDuplex.FO10000MbpsFullDuplex	FO 10000 Mbps Vollduplex
TransmissionRateAndDuplex.FO100MbpsFullDuplexLD	FO 100 Mbps Vollduplex LD
TransmissionRateAndDuplex.POFPCF100MbpsFullDuplex	POF/PCF 100 Mbps Vollduplex

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

5.6 Funktionen für Bibliotheken

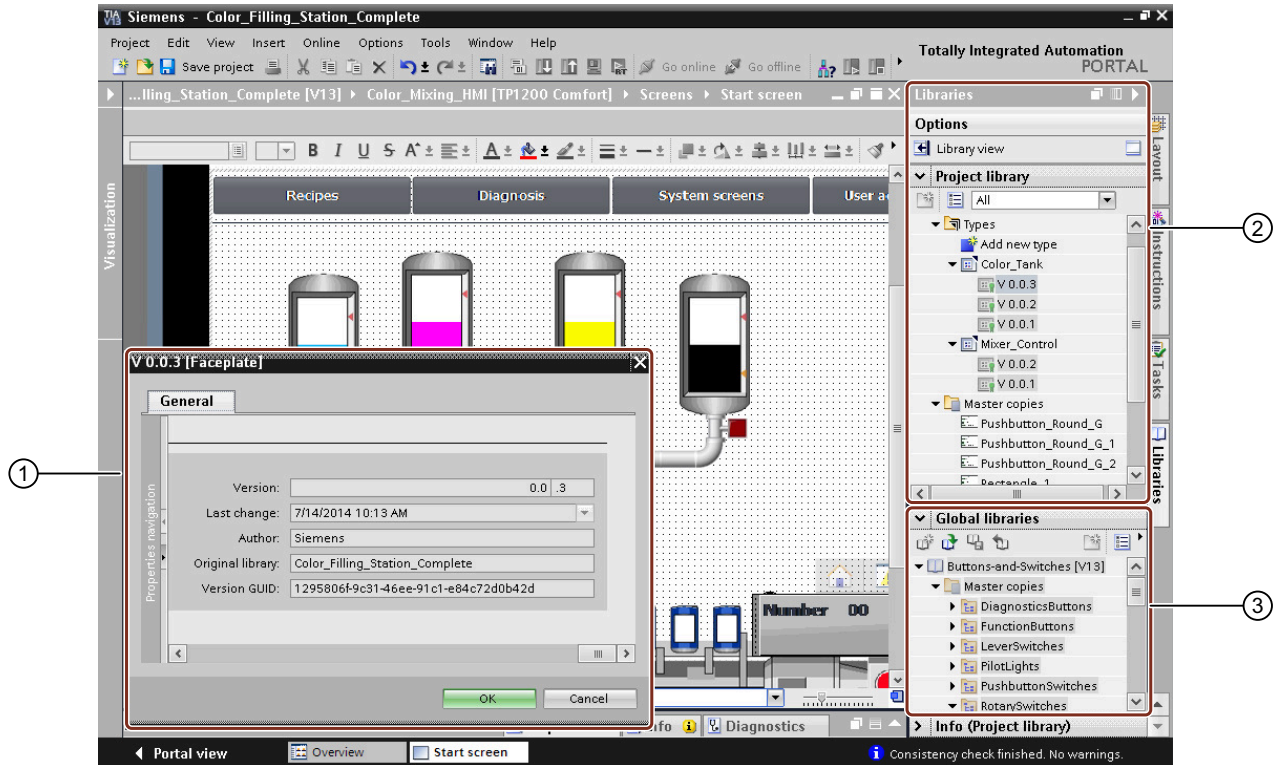
5.6.1 Funktionen auf Objekte und Instanzen

Zugriff auf Typen und Instanzen

Mit der Schnittstelle TIA Portal Openness API können Sie auf Typen, Typversionen und Masterkopien in der Projektbibliothek oder in globalen Bibliotheken zugreifen. Sie können Verbindungen zwischen Typversionen und Instanzen bestimmen. Sie können auch Instanzen im Projekt aktualisieren und Änderungen zwischen einer globalen Bibliothek und der Projektbibliothek synchronisieren. Die Schnittstelle TIA Portal Openness API unterstützt auch den Vergleich von Typversionen und Instanzen.

Funktionen für Objekte und Instanzen

Mit der Schnittstelle TIA Portal Openness API haben Sie Zugriff auf die folgenden Funktionen für Typen, Typversionen, Masterkopien und Instanzen:



① Attribute von Typen, Typversionen, Masterkopien und Instanzen anzeigen

② Die folgenden Funktionen sind in der Projektbibliothek verfügbar:

- Instanzen von Typen aktualisieren
- Typversionen im Projekt instanziiieren
- Innerhalb der Bibliotheksgruppe navigieren
- Gruppe, Typen, Typversionen und Masterkopien löschen

③ Die folgenden Funktionen sind in der globalen Bibliothek verfügbar:

- Instanzen von Typen aktualisieren
- Typversion im Projekt instanziiieren
- Innerhalb der Bibliotheksgruppe navigieren

5.6.2 Auf globale Bibliotheken zugreifen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden. Siehe Verbindung zum TIA Portal aufbauen (Seite 84)

Verwendung

Es gibt drei Arten von globalen Bibliotheken.

- Globale Systembibliothek (Siemens.Engineering.Library.SystemGlobalLibrary): Diese globalen Bibliotheken sind Bestandteil der Installation des TIA Portals und verwenden die Dateierweiterung *.asx. Alle globalen Systembibliotheken sind schreibgeschützt.
- Globale Unternehmensbibliothek (Siemens.Engineering.Library.CorporateGlobalLibrary): Diese globalen Bibliotheken wurden von einem Administrator ausgewählt und werden beim Starten des TIA Portals vorab geladen. Alle globalen Unternehmensbibliotheken sind schreibgeschützt.
- Globale Anwenderbibliothek (Siemens.Engineering.Library.UserGlobalLibrary): Diese globalen Bibliotheken wurden von Benutzern des TIA Portals erstellt. Globale Anwenderbibliotheken können entweder nur schreibgeschützt oder mit Lese- und Schreibrechten geöffnet werden.
Wenn eine globale Anwenderbibliothek in einem bestimmten Modus bereits geöffnet ist, kann diese globale Anwenderbibliothek nicht in einem anderen Modus geöffnet werden. Globale Anwenderbibliotheken aus Vorgängerversionen können nur schreibgeschützt geöffnet werden.

Mit TIA Portal Openness geöffnete globale Bibliotheken werden außerdem der Sammlung der globalen Bibliotheken der TIA Portal-Benutzeroberfläche hinzugefügt und in der Oberfläche im TIA Portal angezeigt, sofern die Oberfläche vorhanden ist.

Programmcode: Verfügbare globale Bibliotheken

Um Informationen über sämtliche globale Bibliotheken abzurufen, ändern Sie folgenden Programmcode:

```
TiaPortal tia = ...;
var availableLibraries = tia.GlobalLibraries.GetGlobalLibraryInfos();
foreach (GlobalLibraryInfo info in availableLibraries)
{
    Console.WriteLine(" Library Name: {0}", info.Name);
    Console.WriteLine(" Library Path: {0}", info.Path);
    Console.WriteLine(" Library Type: {0}", info.LibraryType);
    Console.WriteLine(" Library IsOpen: {0}", info.IsOpen);
}
```

Attribute von GlobalLibrary

Wert	Datentyp	Beschreibung
Author	String	Autor der globalen Bibliothek.
Comment	MultilingualText	Kommentar der globalen Bibliothek.
IsReadOnly	Boolean	Wahr, wenn die globale Bibliothek schreibgeschützt ist.

Wert	Datentyp	Beschreibung
IsModified	Boolean	Wahr, wenn der Inhalt der globalen Bibliothek geändert wurde.
Name	String	Name der globalen Bibliothek.
Path	FileInfo	Pfad der globalen Bibliothek.

Attribute von GlobalLibraryInfo

Wert	Rückgabotyp	Beschreibung
IsReadOnly	Boolean	Wahr, wenn die globale Bibliothek schreibgeschützt ist.
IsOpen	Boolean	Wahr, wenn die globale Bibliothek bereits geöffnet ist.
LibraryType	GlobalLibraryType	Typ der globalen Bibliothek: • System: Globale Systembibliothek • Corporate: Globale Unternehmensbibliothek • User: Globale Anwenderbibliothek
Name	String	Name der globalen Bibliothek.
Path	FileInfo	Pfad der globalen Bibliothek.

Siehe auch

Auf Ordner in einer Bibliothek zugreifen (Seite 224)

5.6.3 Zugriff auf Bibliotheksbasistypen

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness auf die Klassen von Basistypen des Bibliothekstyp-Objekts zugreifen. Diese verfügen alle über einen direkten Basistyp vom Typ `Siemens.Engineering.Library.Types.LibraryType`

Sie können Zugriff auf die unten stehenden Klassen von Basistypen erhalten:

- `Siemens.Engineering.Hmi.Faceplate.FaceplateLibraryType`
- `Siemens.Engineering.Hmi.RuntimeScripting.VBScriptLibraryType`
- `Siemens.Engineering.Hmi.RuntimeScripting.CScriptLibraryType`

- Siemens.Engineering.Hmi.Screen.ScreenLibraryType
- Siemens.Engineering.Hmi.Screen.StyleLibraryType
- Siemens.Engineering.Hmi.Screen.StyleSheetLibraryType
- Siemens.Engineering.Hmi.Tag.HmiUdtLibraryType
- Siemens.Engineering.SW.Blocks.CodeBlockLibraryType
- Siemens.Engineering.SW.Types.PlcTypeLibraryType

Programmcode

```
TiaPortal tiaPortal = ...;
Project project = tia.Projects.First();
ProjectLibrary library = project.ProjectLibrary;
//Pre-V14SP1
LibraryType libraryType= library.TypeFolder.Types.Find("VBFunction_1");
//V14SP1
VBScriptLibraryType libraryTypeAsVbScript = libraryType as VBScriptLibraryType;
```

5.6.4 Zugriff auf Sprachen der globalen Bibliothek

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Eine Bibliothek ist geöffnet.
Siehe Bibliotheken öffnen (Seite 211)

Anwendung

Sie können mithilfe des Navigators für die Spracheinstellungen auf die Sprachen der globalen Bibliothek zugreifen und sie verwalten.

TIA Portal Openness unterstützt den folgenden Zugriff auf die Sprachen der globalen Bibliothek:

- In unterstützten Sprachen iterativ suchen
- Suchen in der Sammlung unterstützter Sprachen mithilfe von `System.Globalization.CultureInfo`.
- Zugriff auf einzelne Sprachen. Jedes Sprachobjekt enthält ein einziges schreibgeschütztes Attribut `Culture` vom Typ `System.Globalization.CultureInfo`.
- Zugriff auf eine Sammlung aktiver Sprachen.
- Suchen in der Sammlung aktiver Sprachen mithilfe von `System.Globalization.CultureInfo`.

- Hinzufügen einer Sprache zu einer Sammlung aktiver Sprachen.
- Entfernen einer Sprache aus einer Sammlung aktiver Sprachen.
- Festlegen einer Bearbeitungssprache.
- Festlegen einer Referenzsprache.

Attribut von Sprachen der globalen Bibliothek

Sprachen der globalen Bibliothek haben die folgenden Attribute:

Attributname	Datentyp	Schreibbar	Beschreibung
LanguageSettings	Siemens.Engineering.LanguageSettings	schr.-gesch.	Verwaltet Sprachen der globalen Bibliothek

Programmcode: Sprachen der globalen Bibliothek enumerieren

Ändern Sie den folgenden Programmcode, um die Sprache der globalen Bibliothek zu enumerieren:

```
FileInfo m_GlobalLibraryPath = new FileInfo("bla");
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
var globalLibrary = portal.GlobalLibraries.Open(m_GlobalLibraryPath, OpenMode.ReadOnly);
LanguageSettings languageSettings = globalLibrary.LanguageSettings;
```

Programmcode: Auf Spracheinstellung zugreifen

Ändern Sie den folgenden Programmcode, um auf die Spracheinstellung der globalen Bibliothek zuzugreifen:

```
FileInfo m_GlobalLibraryPath = new FileInfo("bla");
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
var globalLibrary = portal.GlobalLibraries.Open(m_GlobalLibraryPath, OpenMode.ReadOnly);
LanguageComposition languages = globalLibrary.LanguageSettings.Languages;
foreach (Language language in languages)
{
    // Work with this language
}
```

Programmcode: Festlegen der Sprachen der globalen Bibliothek

Ändern Sie den folgenden Programmcode, um Sprachen der globalen Bibliothek festzulegen. Wenn Sie über TIA Portal Openness eine neue unterstützte Sprache hinzufügen, wird die Sprache der Sammlung aktiver Sprachen hinzugefügt.

```
FileInfo m_GlobalLibraryPath = new FileInfo("bla");
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
var globalLibrary = portal.GlobalLibraries.Open(m_GlobalLibraryPath, OpenMode.ReadOnly);
LanguageSettings languageSettings = globalLibrary.LanguageSettings;
LanguageComposition supportedLanguages = languageSettings.Languages;
LanguageAssociation activeLanguages = languageSettings.ActiveLanguages;
Language supportedGermanLanguage = supportedLanguages.Find(CultureInfo.GetCultureInfo("de-DE"));
activeLanguages.Add(supportedGermanLanguage);
languageSettings.EditingLanguage = supportedGermanLanguage;
languageSettings.ReferenceLanguage = supportedGermanLanguage;
```

Hinweis

Wenn Sie der globalen Bibliothek Sprachen hinzufügen oder sie darin ändern, bewirkt dies keine Änderungen an Sprachen von veröffentlichten Versionen der globalen Bibliothek. Dies gilt auch für die Aktualisierung von Sprachen der globalen Bibliothek über die Benutzeroberfläche (Spracheditor) oder den Navigator für die Spracheinstellungen (LanguageSettings).

5.6.5 Bibliotheken öffnen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet. Diese Voraussetzung gilt nur für den Zugriff auf Projektbibliotheken.
Siehe Projekt öffnen (Seite 131)

Verwendung

Eine globale Bibliothek kann über System.IO.FileInfo mit einem Pfad auf die Bibliotheksdatei auf einem lokalen Speichermedium oder einem Netzwerkspeicher geöffnet werden. Nur globale Anwenderbibliotheken können mittels Pfad geöffnet werden. Ein Pfad einer globalen Systembibliothek oder einer globalen Unternehmensbibliothek kann zum Öffnen der Bibliothek nicht verwendet werden.

Ab V14 SP1 können globale Bibliotheken mittels GlobalLibraryInfo geöffnet werden. Der OpenMode wird in der GlobalLibraryInfo angegeben.

Eine globale Anwenderbibliothek einer Vorgängerversion des TIA Portals kann hochgerüstet und mit der aktuellen Version des TIA Portals geöffnet werden. Eine globale Bibliothek der V13 oder einer Vorgängerversion kann nicht mittels Hochrüsten geöffnet werden. Diese Bibliotheken müssen zunächst auf V13 SP1 aktualisiert werden.

Mit TIA Portal Openness geöffnete globale Bibliotheken werden außerdem der Sammlung der globalen Bibliotheken im TIA Portal hinzugefügt und in der Benutzeroberfläche im TIA Portal angezeigt.

Programmcode: Bibliothek mittels System.IO.FileInfo öffnen

Ändern Sie folgenden Programmcode:

```
TiaPortal tia = ...
FileInfo fileInfo = ....

UserGlobalLibrary userLib = tia.GlobalLibraries.Open(fileInfo, OpenMode.ReadWrite);
```

Programmcode: Bibliothek mittels GlobalLibraryInfo öffnen

Ändern Sie folgenden Programmcode:

```
TiaPortal tia = ...
IList<GlobalLibraryInfo> libraryInfos = tia.GlobalLibraries.GetGlobalLibraryInfos();
GlobalLibraryInfo libInfo = ...; //check for the info you need from the list, e.g.
GlobalLibrary libraryOpenedWithInfo;
if (libInfo.Name == "myLibrary")
    libraryOpenedWithInfo = tia.GlobalLibraries.Open(libInfo);
```

Programmcode: Bibliothek hochrüsten

Ändern Sie folgenden Programmcode:

```
TiaPortal tia = ...
FileInfo fileInfo = .... //library from previous TIA Portal version

UserGlobalLibrary userLib = tia.GlobalLibraries.OpenWithUpgrade(fileInfo);
```

OpenMode

Wert	Beschreibung
ReadOnly	Schreibzugriff auf die Bibliothek.
ReadWrite	Lese- und Schreibzugriff auf die Bibliothek.

5.6.6 Bibliotheken vergleichen

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Das Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Mi Hilfe von TIA Portal Openness können Sie zwei Bibliotheken miteinander vergleichen. Sie können dieses Feature beispielsweise nutzen, um die Unterschiede zwischen Versionen zu ermitteln, bevor Sie ein Projekt mit der neuen Version aktualisieren.

Mi Hilfe von Openness können Sie einen detaillierten Vergleich ausgewählter Bibliotheksobjekte vornehmen. Hierbei können zwei Bibliotheksobjekte verglichen werden.

- Typobjekt
- Versionsobjekt
- Masterkopien

Die Openness API ermöglicht den Vergleich zweier Typobjekte, z. B. eines Versionsobjekts oder von Masterkopien, und gibt ein detailliertes Ergebnis des Inhaltsvergleichs zurück.

Methode, Parameter und Enum-Werte

TIA Portal Openness unterstützt die folgende Methode mit ihrem Signaturtyp für den Vergleich von Bibliotheksobjekten:

Methode	Rückgabotyp
DetailedCompareResult CompareTo(LibraryType targetLibraryObject)	Siemens.Engineering.Library.Types.LibraryType
DetailedCompareResult CompareTo(LibraryTypeVersion targetLibraryObject)	Siemens.Engineering.Library.Types.LibraryTypeVersion
DetailedCompareResult CompareTo(MasterCopy targetLibraryObject)	Siemens.Engineering.Library.MasterCopies.MasterCopy

Der Rückgabotyp LibraryCompareResult unterstützt den folgenden Navigator in TIA Portal Openness:

Name	Typen
Parent	Siemens.Engineering.IEngineeringObject
RootElement	Siemens.Engineering.Library.Compare.LibraryCompareResultElement

Der Rückgabetyt LibraryCompareResultElement unterstützt den folgenden Navigator- und Attributnamen in TIA Portal Openness:

Name	Typen
Parent	Siemens.Engineering.Library.Compare.LibraryCompareResultElementComposition
RootElement	Siemens.Engineering.Library.Compare.LibraryCompareResultElement

Attributname	Typ
ComparisionResult	Siemens.Engineering.Library.Compare.LibraryCompareResultState
DetailedInformation	System.String
LeftName	System.String
RightName	System.String
Left	Siemens.Engineering.IEngineeringObject
Right	Siemens.Engineering.IEngineeringObject

Die ENUM LibraryCompareResultState unterstützt die folgenden ENUM-Elemente:

Enum-Elemente	Beschreibung
ContainerContentsDifferent	Container contents has one or more differences
ContainerContentsIdentical	Container content is identical
ObjectsDifferent	Objects are different
LeftMissing	Left object is missing
RightMissing	Right object is missing
ObjectsIdentical	Objects are identical

DetailedCompareResult unterstützt den folgenden Navigator in TIA Portal Openness:

Name	Typ
Parent	Siemens.Engineering.IEngineeringObject
RootElement	Siemens.Engineering.Library.Compare.LibraryCompareResultElement

Der Rückgabetyt DetailedCompareResultElement unterstützt den folgenden Navigator- und Attributnamen in TIA Portal Openness:

Name	Typ
Parent	Siemens.Engineering.IEngineeringObject

Attributname	Typ	Beschreibung
Description	System.String	Detailvergleich des Eigenschaftsnamens
LeftValue	System.Object	Detailvergleich des linken Eigenschaftswertes. Die Eigenschaft LeftValue unterstützt die folgenden Typen, nämlich System.Guid, System.Boolean, System.String, System.DateTime
RightValue	System.Object	Detailvergleich des rechten Eigenschaftswertes. Die Eigenschaft RightValue unterstützt die folgenden Typen, nämlich System.Guid, System.Boolean, System.String, System.DateTime.
DetailCompare-Status	Siemens.Engineering.Library.Compare.DetailCompareStatus	Gibt das Vergleichsergebnis der Eigenschaftswerte anhand der angewandten standardmäßigen Kriterienfilter zurück. (Die Eigenschaft der Originalbibliothek des Versionsobjekts wird nicht standardmäßig verglichen)

DetailCompareStatus unterstützt die folgenden ENUM-Elemente:

Enum-Elemente	Wert
ObjectsDifferent	Objekte sind verschieden
ObjectsIdentical	Objekte sind identisch
NotCompared	Bei Anwendung von Filtern werden Objekte nicht verglichen

Programmcode

```
...
GlobalLibrary globalLibrary = ...;
ProjectLibrary projectLibrary = Project.ProjectLibrary;
LibraryCompareResult libraryCompareResults =
globalLibrary.CompareToLibrary(projectLibrary);
...
```

Um zwei Typobjekte zu vergleichen, ändern Sie folgenden Programmcode:

```
...
GlobalLibrary globalLibrary = ...;
ProjectLibrary projectLibrary = Project.ProjectLibrary;
LibraryType leftTypeObject = ProjectLib.TypeFolder.Types.Find(typeName);
LibraryType rightTypeObject = GlobalLib.TypeFolder.Types.Find(typeName);
DetailedCompareResult detailedCompareResult = leftTypeObject.CompareTo(rightTypeObject);
...
```

Um zwei Typobjekte zu vergleichen, ändern Sie folgenden Programmcode:

```
...
GlobalLibrary globalLibrary = ...;
ProjectLibrary projectLibrary = Project.ProjectLibrary;
LibraryTypeVersion leftVersionObject =
ProjectLib.TypeFolder.Types.Find(typeName).Versions[0];
LibraryTypeVersion rightVersionObject =
GlobalLib.TypeFolder.Types.Find(typeName).Versions[0];
DetailedCompareResult detailedCompareResult =
leftVersionObject.CompareTo(rightVersionObject);
...
```

Um zwei Masterkopienobjekte zu vergleichen, ändern Sie folgenden Programmcode:

```
...
GlobalLibrary globalLibrary = ...;
ProjectLibrary projectLibrary = Project.ProjectLibrary;
LibraryTypeVersion leftMasterCopy =
ProjectLib.MasterCopyFolder.MasterCopies.Find(masterCopyName);
LibraryTypeVersion rightMasterCopy =
GlobalLib.MasterCopyFolder.MasterCopies.Find(masterCopyName);
DetailedCompareResult detailedCompareResult = leftMasterCopy.CompareTo(rightMasterCopy);
...
```

5.6.7 Typ aus Versionsobjekt erstellen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness einen Typ aus dem Versionsobjekt erstellen. Diese API-Funktion ist über den Vorgang CreateType innerhalb der LibraryTypeComposition verfügbar.

Methode und Parameter

CreateFrom() ist eine Überladungs-API-Methode, die die folgenden Parameter akzeptiert und als Datentyp LibraryType zurückgibt.

Parametername	Rückgabetyt	Beschreibung
sourceVersion	LibraryTypeVersion	Gibt die Quelle an, aus der der Benutzer einen Typ erstellen möchte
typeName	String	Gibt einen Namen für den neu erstellten Typ an. Die API prüft dann, ob der vom Benutzer übergebene Wert nicht null und nicht leer ist. Wenn der bereitgestellte Wert diese Kriterien erfüllt, wird er als Name für den Typ verwendet.
typeMtdv	String	Gilt für den Faceplate-Typ 'Unified', der die Minimum Target Device Version (MTDV) aufweist. Die API ignoriert den vom Benutzer übergebenen Parameter typeMTDV, wenn der Typ nicht der Faceplate-Typ 'Unified' ist. Sie können einen neuen Typ der MinimumTargetDeviceVersion aus einem Quelltyp erstellen: - Wenn der vom Benutzer bereitgestellte Parameter typeMtdv als MTDV für den neu erstellten Typ zugewiesen wird. Diese Zuweisung erfolgt jedoch nur dann, wenn der in typeMtdv bereitgestellte Wert ein unterstützter MTDV-Wert ist. - Wenn der vom Benutzer bereitgestellte Parameter typeMtdv null oder leer ist, ist der MTDV-Typ des neu erstellten Typs identisch mit dem Quelltyp.

Programmcode: Erstellen des Typs mit LibraryTypeComposition

Ändern Sie folgenden Programmcode, um einen neuen Typ der Minimum Target Device Version aus einem Quelltyp zu erstellen:

```
private static void createTypeFromVersionObject(project Project)
{
    //...
    ProjectLibrary projectLibrary = Project.ProjectLibrary;
    LibraryTypeComposition projectLibraryTypes = project.ProjectLibrary.TypeFolder.Types;
    LibraryType sourceLibraryType = projectLibraryTypes[nameofthetype];
    LibraryTypeVersion sourceLibraryTypeVersion = sourceLibraryType.Version[index];
    LibraryType createType = projectLibraryTypes.CreateFrom(sourceLibraryTypeVersion);
    //Or
    LibraryType createType = projectLibraryTypes.CreateFrom(sourceLibraryTypeVersion, typeName);
    //Or
    LibraryType createType = projectLibraryTypes.CreateFrom(sourceLibraryTypeVersion, typeMtdv);
    //Or
    LibraryType createType =
    projectLibraryTypes.CreateFrom(sourceLibraryTypeVersion, typeMtdv, typeName);
    //...
}
```

Ausnahme

Die `Siemens.Engineering.EngineeringTargetInvocationException` wird für den Benutzer in folgenden Fällen ausgelöst:

- Um einen `CreateFrom`-Vorgang mit einem `LibraryTypeVersion`-Objekt 'null' durchzuführen
- Um einen `CreateFrom`-Vorgang mit einem `LibraryTypeVersion`-Objekt durchzuführen, das sich in der globalen Bibliothek befindet
- Um einen `CreateFrom`-Vorgang mit einem schreibgeschützten `LibraryTypeVersion`-Objekt durchzuführen
- Um einen `CreateFrom`-Vorgang mit einem `LibraryTypeVersion`-Objekt durchzuführen, das sich bereits 'in Arbeit' oder 'im Test' befindet
- Um einen `CreateFrom`-Vorgang mit einem `LibraryTypeVersion`-Objekt mit einer nicht unterstützten `MinimumTargetDeviceVersion` (MTDV) durchzuführen
- Um einen `CreateFrom`-Vorgang mit einer `LibraryTypeComposition` einer `GlobalLibrary` durchzuführen

5.6.8 Offene Bibliotheken enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)

Anwendung

Alle geöffneten globalen Bibliotheken im TIA Portal, unabhängig davon, ob sie über API oder die Benutzeroberfläche geöffnet wurden, können enumeriert werden.

Globale Bibliotheken aus Vorgängerversionen des TIA Portals werden nicht enumeriert, wenn sie mit Schreibzugriff geöffnet wurden.

Programmcode

Ändern Sie den folgenden Programmcode, um offene globale Bibliotheken zu enumerieren:

```
TiaPortal tia = ...
foreach (GlobalLibrary globLib in tia.GlobalLibraries)
{
    ///work with the global library
}
```

Siehe auch

Projekt öffnen (Seite 131)

5.6.9 Bibliotheken speichern und schließen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Eine Bibliothek ist geöffnet.
Siehe Bibliotheken öffnen (Seite 211)

Anwendung

Globale Anwenderbibliotheken können geschlossen oder gespeichert werden. Alle an der globalen Bibliothek vorgenommenen Änderungen werden nicht automatisch gespeichert. Alle nicht gespeicherten Änderungen werden ohne Benutzeraufforderung durch Schließen einer globalen Bibliothek verworfen.

Globale Systembibliotheken und globale Unternehmensbibliotheken können nicht geschlossen oder gespeichert werden.

Zum Speichern und Schließen einer globalen Bibliothek:

- Zum Speichern einer globalen Anwenderbibliothek verwenden Sie die Methode `Save()`.
- Verwenden Sie die Methode `SaveAs()`, wenn Sie eine globale Anwenderbibliothek in einem anderen Verzeichnis speichern möchten.
- Zum Schließen einer globalen Anwenderbibliothek verwenden Sie die Methode `Close()`.

Feature-Token für `SaveAs()` sind:

- `Public API`: Erforderlich für alle veröffentlichten Features
- `DenyIfTransaction`: Erforderlich, da `SaveAs` nicht innerhalb einer Transaktion zulässig sein darf, da es nicht rückgängig gemacht werden kann.

Programmcode

Ändern Sie folgenden Programmcode, um eine globale Anwenderbibliothek zu speichern:

```
UserGlobalLibrary userLib = ...  
// save changes and close library  
userLib.Save();  
userLib.Close();
```

Ändern Sie folgenden Programmcode, um eine globale Anwenderbibliothek an einem anderen Ort zu speichern:

```
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
GlobalLibraryComposition globalLibraryComposition = portal.GlobalLibraries;
//please adapt the path and the extension alx to the installed version of TIA Portal
FileInfo existingLibraryFileInfo = new
FileInfo(@"D:\GlobalLibraries\MyGlobalLibrary\MyGlobalLibrary.alx");
DirectoryInfo targetDirectoryInfo = new
DirectoryInfo(@"D:\GlobalLibraries\GlobalLibrarySaveAs");
UserGlobalLibrary userGlobalLibrary =
globalLibraryComposition.Open(existingLibraryFileInfo, OpenMode.ReadWrite);
userGlobalLibrary.SaveAs(targetDirectoryInfo);
```

Hinweis

Die Beständigkeit der Bibliothek wird nach der Operation SaveAs geändert. Von daher ist die neu gespeicherte Bibliothek nach Durchführung von SaveAs in der Bibliothekskarte verfügbar. Die ursprüngliche Bibliothek jedoch, mit der die Operation SaveAs durchgeführt wurde, ist nicht verfügbar. Die neu gespeicherte Bibliothek hat den gleichen Modus wie die ursprüngliche Bibliothek, für die SaveAs durchgeführt wurde. Das bedeutet, dass die neu gespeicherte Bibliothek im Zielpfad auch im Modus ReadOnly geöffnet wird, wenn die vorhandene Bibliothek im Modus ReadOnly geöffnet wurde. Ähnliches gilt für den Modus ReadWrite.

Ändern Sie folgenden Programmcode, um eine globale Anwenderbibliothek zu schließen:

```
UserGlobalLibrary userLib = ...
// close and discard changes
userLib.Close();
```

5.6.10 Bibliothek archivieren und abrufen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Eine Bibliothek ist geöffnet.
Siehe Bibliotheken öffnen (Seite 211)
- Eine Bibliothek ist gespeichert
Siehe Bibliotheken speichern und schließen (Seite 219)

Einleitung

Sie können mit TIA Portal Openness geöffnete und gespeicherte globale Bibliotheken archivieren, bevor weitere Änderungen daran vorgenommen werden. Auf diese Weise verhindern Sie unbeabsichtigte Änderungen und können die archivierte Bibliothek später wieder über TIA Portal Openness abrufen. Außerdem können Sie die archivierte Datei problemlos im Netzwerk teilen.

Bibliothek archivieren

Über die Schnittstelle TIA Portal Openness API können Sie eine globale Anwenderbibliothek archivieren. Die API ist am Objekt `Siemens.Engineering.UserGlobalLibrary` verfügbar.

Die API-Definition lautet wie folgt:

```
public void Archive(System.IO.DirectoryInfo targetDirectory, string targetName,
Siemens.Engineering.Library.LibraryArchivationMode
archivationMode)
```

Hierbei ist `targetName` der Name der Datei, die für die archivierten oder nicht-archivierten Bibliotheken erstellt wurde. Diese Datei kann Dateierweiterungen oder keine Dateierweiterungen enthalten. Wenn Sie keine Erweiterung oder eine andere Erweiterung als "zalx" oder "zal14" usw. angeben, kann die archivierte Datei nicht außerhalb der Openness API aus dem TIA Portal abgerufen werden.

Wenn der Wert für `LibraryArchivationMode` mit `Compressed` und `DiscardRestorableDataAndCompressed` angegeben wird, ist der Name der archivierten Datei identisch mit dem von Ihnen bereitgestellten Namen. Bei den Werten `None` und `DiscardRestorableData` für `LibraryArchivationMode` wird die Erweiterung der Bibliotheksdatei automatisch von der Komponente Project Manager des TIA Portals anhand der aktuellen Version des TIA Portals festgelegt.

Hinweis

Sie müssen die Bibliothek vor dem Aufrufen der Archivierungs-API speichern. Falls die Bibliothek nicht gespeicherte Änderungen enthält, löst das Archiv eine `EngineeringTargetInvocationException` aus.

Archivierungsmodus der Bibliothek

Für die Enumeration `LibraryArchivationMode` (Archivierungsmodus der Bibliothek) können vier Werte angegeben werden.

LibraryArchivation-Mode	Beschreibung
None	- Mit den Originaldateien werden keine speziellen Aktionen durchgeführt. Der Modus ähnelt dem Vorgang "Speichern unter". - In diesem Modus wird keine komprimierte ZIP-Datei erstellt. - Der Unterschied zu <code>SaveAs</code> besteht in diesem Fall darin, dass das Archiv den persistenten Speicherort nicht auf den neuen Ordner <code>Archived</code> umstellt, was jedoch bei <code>SaveAs</code> der Fall ist.
DiscardRestorableData	- Bei der Dateispeicherung wird die Bibliothek in einer internen Datendatei abgelegt. Diese Datei wird anschließend bei jeder Änderung an den Bibliotheksdaten größer. Beim Modus <code>DiscardRestorableData</code> wird diese Datendatei reorganisiert (dabei wird nur die neueste Version der Objekte gespeichert und der Verlauf aus der Datei gelöscht). Zwischenzeitlich angefallene Daten, die Dateien des IM-Verzeichnisses und des Verzeichnisses <code>tmp</code> (siehe Struktur der Bibliotheksverzeichnisse) werden nicht an den Archiv-Speicherort kopiert. - In diesem Modus wird keine komprimierte ZIP-Datei erstellt.
Compressed	Die vom Archivierungsprozess erstellte Struktur des TMP-Bibliotheksordners wird zu einem ZIP-kompatiblen Archiv komprimiert. Nach Erstellung der ZIP-Datei wird die TMP-Ordnerstruktur gelöscht.
DiscardRestorableDataAndCompressed	Die vom Archivierungsprozess erstellte Struktur des TMP-Bibliotheksordners verwirft die wiederherstellbaren Daten und wird dann zu einem ZIP-kompatiblen Archiv komprimiert. Nach Erstellung der ZIP-Datei wird die TMP-Ordnerstruktur gelöscht.

Programmcode: Bibliothek archivieren

Ändern Sie folgenden Programmcode, um eine globale Anwenderbibliothek zu archivieren:

```
var tiaPortal = new TiaPortal(TiaPortalMode.WithoutUserInterface);
//Please adapt the path and the extension alx to the installed version of TIA Portal
var libraryFilePath = @"E:\Sample1\Sample1.alx";
var userGlobalLibrary = tiaPortal.GlobalLibraries.Open(new FileInfo(libraryFilePath),
OpenMode.ReadWrite);
var archivePath = @"E:\Archive";
var archiveFileName = "SampleArchive";
userGlobalLibrary.Archive(new DirectoryInfo(archivePath), archiveFileName,
LibraryArchivationMode.Compressed);
```

Bibliothek abrufen

Über die Schnittstelle TIA Portal Openness API können Sie eine archivierte TIA Portal-Bibliothek abrufen. Der Abruf ist nur mit einem komprimierten Archiv möglich. Die API ist am Objekt `Siemens.Engineering.GlobalLibraryComposition` verfügbar.

Die API-Definition lautet wie folgt:

```
public Siemens.Engineering.Library.UserGlobalLibrary Retrieve(System.IO.FileInfo
sourcePath, System.IO.DirectoryInfo targetDirectory, Siemens.Engineering.OpenMode openMode)
```

Hinweis

Es ist nicht möglich, archivierte Bibliotheken mit den Enumerationswerten 'LibraryArchivationMode.None' oder 'LibraryArchivationMode.DiscardRestorableData' abzurufen.

Durch Aufrufen von `RetrieveWithUpgrade` API stellen Sie die archivierte Bibliothek einer früheren TIA Portal-Version wieder her. Die API-Definition lautet wie folgt:

```
public Siemens.Engineering.Library.UserGlobalLibrary  
RetrieveWithUpgrade(System.IO.FileInfo sourcePath, System.IO.DirectoryInfo  
targetDirectory, Siemens.Engineering.OpenMode openMode)
```

Öffnungsmodus

Die Enumeration `OpenMode` weist zwei Werte auf:

OpenMode	Beschreibung
ReadMode	Lesezugriff auf die Bibliothek. Daten können aus der Bibliothek gelesen werden.
ReadWrite	Schreibzugriff auf die Bibliothek. Daten können in die Bibliothek geschrieben werden.

Programmcode: Bibliothek abrufen

Ändern Sie den folgenden Programmcode, um eine Bibliothek abzurufen:

```
//Please adapt the path and the extension zalx to the installed version of TIA Portal  
var archivePath = @"E:\Archive\Sample1.zalx";  
var retrievedLibraryDirectory = @"E:\RetrievedLibraries";  
var tiaPortal = new TiaPortal(TiaPortalMode.WithoutUserInterface);  
tiaPortal.GlobalLibraries.Retrieve(new FileInfo(archivePath), new  
DirectoryInfo(retrievedLibraryDirectory), OpenMode.ReadWrite));
```

Siehe auch

[Bibliotheken öffnen \(Seite 211\)](#)

5.6.11 Globale Bibliotheken erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe [Verbindung zum TIA Portal aufbauen \(Seite 84\)](#)

Einleitung

Globale Bibliotheken können über die TIA Portal Openness API erzeugt werden, indem die Methode `Create` auf `GlobalLibraryComposition` aufgerufen wird. Es wird eine globale Anwenderbibliothek ausgegeben.

Parameter zum Erstellen globaler Bibliotheken

Parameter	Datentyp	Typ	Beschreibung
Author	String	Obligatorisch	Autor einer globalen Bibliothek.
Comment	String	Optional	Kommentar einer globalen Bibliothek.
Name	String	Optional	Name einer globalen Bibliothek.
TargetDirectory	DirectoryInfo	Obligatorisch	Verzeichnis, das den Ordner mit der globalen Bibliothek enthalten wird.

GlobalLibraryComposition.Create

Ändern Sie folgenden Programmcode:

```
TiaPortal tia= ...;  
DirectoryInfo targetDirectory = new DirectoryInfo(@"D:\GlobalLibraries");  
UserGlobalLibrary globalLibrary =  
tia.GlobalLibraries.Create<UserGlobalLibrary>(targetDirectory, "Library1")
```

Entsprechend diesem Beispiel:

- wird ein Ordner "D:\GlobalLibraries\Library1" erstellt
- wird eine globale Bibliotheksdatei "D:\GlobalLibraries\Library1\Library1.alXX" erstellt

Siehe auch

Projekt öffnen (Seite 131)

5.6.12 Auf Ordner in einer Bibliothek zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek.
Siehe Auf globale Bibliotheken zugreifen (Seite 206)

Einleitung

Über die Schnittstelle TIA Portal Openness API können Sie auf die Systemordner für Typen und Masterkopien in einer Bibliothek zugreifen. Dabei können Sie auf Typen, Typversionen, Masterkopien und benutzerdefinierte Ordner im Systemordner zugreifen.

Mit der Methode `Find`, z. B.

`libTypeUserFolder.Folders.Find("SomeUserFolder")` ;, können Sie jederzeit auf einen benutzerdefinierten Ordner zugreifen.

Programmcode: Auf Systemordner zugreifen

Um auf die Systemordner für Typen in einer Bibliothek zuzugreifen, ändern Sie folgenden Programmcode:

```
public static void AccessTypeSystemFolder(ILibrary library)
{
    LibraryTypeSystemFolder libTypeSystemFolder = library.TypeFolder;
}
```

Um auf die Systemordner für Masterkopien in einer Bibliothek zuzugreifen, ändern Sie folgenden Programmcode:

```
public static void AccessMasterCopySystemFolder(ILibrary library)
{
    MasterCopySystemFolder libMasterCopySystemFolder = library.MasterCopyFolder;
}
```

Programmcode: Über die Methode `Find()` auf benutzerdefinierte Ordner zugreifen

```
...
LibraryTypeUserFolderComposition userFolderComposition = ...
LibraryTypeUserFolder userFolder = userFolderComposition.Find("Name of user folder");
...
```

Programmcode: Benutzerdefinierte Ordner enumerieren

Um benutzerdefinierte Unterordner in einem Systemordner für Typen zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateUserFoldersInTypeSystemFolder(ILibrary library)
{
    // Enumerating user folders in type system folder:
    LibraryTypeSystemFolder libTypeSystemFolder = library.TypeFolder;
    foreach (LibraryTypeUserFolder libTypeUserFolder in libTypeSystemFolder.Folders)
    {
        //...
    }
}
```

Um benutzerdefinierte Unterordner in einem Systemordner für Masterkopien zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateUserFoldersInMasterCopySystemFolder(ILibrary library)
{
    // Enumerating user folders in master copy system folder:
    MasterCopySystemFolder libMasterCopySystemFolder = library.MasterCopyFolder;
    foreach (MasterCopyUserFolder libMasterCopyUserFolder in
libMasterCopySystemFolder.Folders)
    {
        //..
    }
}
```

Um benutzerdefinierte Unterordner in einem benutzerdefinierten Ordner für Typen zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateAllUserFolders(LibraryTypeUserFolder libUserFolder)
{
    foreach (LibraryTypeUserFolder libSubUserFolder in libUserFolder.Folders)
    {
        EnumerateAllUserFolders(libSubUserFolder);
    }
}
```

Um benutzerdefinierte Unterordner in einem benutzerdefinierten Ordner für Masterkopien zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateAllUserFolders(MasterCopyUserFolder libUserFolder)
{
    foreach (MasterCopyUserFolder libSubUserFolder in libUserFolder.Folders)
    {
        EnumerateAllUserFolders(libSubUserFolder);
    }
}
```

Programmcode: Benutzerdefinierte Ordner erstellen

Um einen benutzerdefinierten Ordner für Typen zu erstellen, ändern Sie folgenden Programmcode:

```
var typeFolderComposition = ProjectLibrary.TypeFolder.Folders;
var newTypeUserFolder = typeFolderComposition.Create("NewTypeUserFolder");
```

Um einen benutzerdefinierten Ordner für Masterkopien zu erstellen, ändern Sie folgenden Programmcode:

```
var masterCopyFolderComposition = projectProjectLibrary.MasterCopyFolder.Folders;
MasterCopyUserFolder newMasterCopyUserFolder =
masterCopyFolderComposition.Create("NewMasterCopyUserFolder");
```

Programmcode: Benutzerdefinierte Ordner umbenennen

Um einen benutzerdefinierten Ordner für Typen zu erstellen, ändern Sie folgenden Programmcode:

```
var typeUserFolder =
project.ProjectLibrary.TypeFolder.Folders.Find("SampleTypeUserFolderName");
typeUserFolder.Name = "NewTypeUserFolderName";
```

```
var typeUserFolder = ProjectLibrary.TypeFolder.Folders.Find("SampleTypeUserFolderName");
typeUserFolder.SetAttributes(new[] {new KeyValuePair<string,object>("Name",
"NewTypeUserFolderName")});
```

Um einen benutzerdefinierten Ordner für Masterkopien zu erstellen, ändern Sie folgenden Programmcode:

```
var masterCopyUserFolder =
project.ProjectLibrary.MasterCopyFolder.Folders.Find("SampleMasterCopyUserFolderName");
masterCopyUserFolder.Name = "NewMasterCopyUserFolderName";
```

```
var masterCopyUserFolder =
ProjectLibrary.MasterCopyFolder.Folders.Find("SampleMasterCopyUserFolderName");
masterCopyUserFolder.SetAttributes(new[] {new KeyValuePair<string,object>("Name",
"NewMasterCopyUserFolderName")});
```

Siehe auch

Auf Kopiervorlagen zugreifen (Seite 239)

5.6.13 Auf Typen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet. Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek. Siehe Auf globale Bibliotheken zugreifen (Seite 206).
- Sie haben Zugriff auf eine Gruppe für Typen. Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 224).

Verwendung

Auf die Typen in einer Bibliothek können Sie über die Schnittstelle TIA Portal Openness API zugreifen.

- Sie können die Typen enumerieren.
- Sie können Typen umbenennen.
- Sie können auf die folgenden Attribute der einzelnen Typen zugreifen:

Attribut	Datentyp	Beschreibung
Author	String	Gibt den Namen des Autors zurück.
Comment	MultilingualText	Gibt den Kommentar zurück.
Guid	Guid	Gibt die GUID des Typs zurück. ¹
Name	String	Gibt den Namen des Typs zurück. ²

¹ Mit diesem Attribut können Sie einen einzelnen Typ in einer Bibliothek finden. Die Suche ist rekursiv.

² Über dieses Attribut können Sie einen einzelnen Typ in einem Ordner finden. Unterordner werden nicht in die Suche einbezogen. Die Typennamen sind nicht eindeutig. Es kann mehrere Typen mit demselben Namen in verschiedenen Gruppen geben. Die GUID des Typs ist hingegen eindeutig.

Unterklassen für Bibliothekstypobjekte

Mit der TIA Portal Openness API können Sie über Unterklassen auf Bibliothekstypobjekte zugreifen. Folgende Unterklassen sind vorhanden:

- Siemens.Engineering.Hmi.Faceplate.FaceplateLibraryType
- Siemens.Engineering.Hmi.Faceplate.FaceplateLibraryTypeVersion
- Siemens.Engineering.Hmi.RuntimeScripting.VBScriptLibraryType
- Siemens.Engineering.Hmi.RuntimeScripting.VBScriptLibraryTypeVersion
- Siemens.Engineering.Hmi.RuntimeScripting.CScriptLibraryType
- Siemens.Engineering.Hmi.RuntimeScripting.CScriptLibraryTypeVersion
- Siemens.Engineering.Hmi.Screen.ScreenLibraryType

- Siemens.Engineering.Hmi.Screen.ScreenLibraryTypeVersion
- Siemens.Engineering.Hmi.Screen.StyleLibraryType
- Siemens.Engineering.Hmi.Screen.StyleLibraryTypeVersion
- Siemens.Engineering.Hmi.Screen.StyleSheetLibraryTypeVersion
- Siemens.Engineering.Hmi.Tag.HmiUdtLibraryType
- Siemens.Engineering.Hmi.Tag.HmiUdtLibraryTypeVersion
- Siemens.Engineering.SW.Blocks.CodeBlockLibraryType
- Siemens.Engineering.SW.Blocks.CodeBlockLibraryTypeVersion
- Siemens.Engineering.SW.Types.PlcTypeLibraryType
- Siemens.Engineering.SW.Types.PlcTypeLibraryTypeVersion

Der folgende Code ist ein Beispiel für die Verwendung von Unterklassen für Bibliothekstypen.

```
ProjectLibrary library = project.ProjectLibrary;
VBScriptLibraryType vbScriptType = ...;
VBScriptLibraryType libraryTypeAsVbScript = libraryType as VBScriptLibraryType;
VBScriptLibraryTypeVersion libraryTypeVersionAsVbScript = libraryTypeVersion as
VBScriptLibraryTypeVersion
```

Programmcode

Um alle Typen im Systemordner einer Bibliothek zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateTypesInTypesSystemFolder (LibraryTypeSystemFolder
libraryTypeSystemFolder)
{
    foreach (LibraryType libraryType in libraryTypeSystemFolder.Types)
    {
        //...
    }
}
```

Um alle Typen in einem benutzerdefinierten Ordner einer Bibliothek zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateTypesInTypesUserFolder (LibraryTypeUserFolder
libraryTypeUserGroup)
{
    foreach (LibraryType libraryType in libraryTypeUserGroup.Types)
    {
        //...
    }
}
```

Um auf die Attribute eines Typs zuzugreifen, ändern Sie folgenden Programmcode:

```
public static void InspectPropertiesOfType (LibraryType libTypeObject)
{
    string typeAuthor = libTypeObject.Author;
    MultilingualText typeComment = libTypeObject.Comment;
    string typeName = libTypeObject.Name;
    Guid typeGUID = libTypeObject.Guid;
}
```

Um einen einzelnen Typ anhand seines Namens oder der GUID zu finden, ändern Sie folgenden Programmcode:

```
public static void FindTypeObjectInLibrary(ILibrary library)
{
    // Find type object by its GUID in a given library:
    System.Guid targetGuid = ...;
    LibraryType libTypeByGUID = library.FindType(targetGuid);
    // Find type object by its name in a given group:
    LibraryTypeFolder libTypeSystemFolder = library.TypeFolder;
    LibraryType libTypeByName = libTypeSystemFolder.Types.Find("myTypeObject");
}
```

Um einen Typ umzubenennen, ändern Sie folgenden Programmcode:

```
// Setting the name attribute
var type = project.ProjectLibrary.TypeFolder.Types.Find("SampleTypeName");
type.Name = "NewTypeName";

//Setting the name attribute dynamically
var type = project.ProjectLibrary.TypeFolder.Types.Find("SampleTypeName");
type.SetAttributes(new[] {new KeyValuePair<string,object>("Name", "NewTypeName")});
```

5.6.14 Auf Typ-Versionen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek.
Siehe Auf globale Bibliotheken zugreifen (Seite 206).
- Sie haben Zugriff auf eine Gruppe für Typen.
Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 224).

Einleitung

Sie können mit TIA Portal Openness folgende Funktionen ausführen:

- Auf Typversionen zugreifen
- Typversionen eines Typs enumerieren
- Typ ermitteln, zu dem eine Typversion gehört
- Instanzen einer Typversion enumerieren
- Neue Instanz einer Typversion erstellen
- Von einer Instanz zu dem damit verbundenen Versionsobjekt navigieren.

Sie können auf die folgenden Attribute der einzelnen Typversionen zugreifen:

Attribut	Datentyp	Beschreibung
Author	String	Gibt den Namen des Autors zurück.
Comment	MultilingualText	Gibt den Kommentar zurück.
Guid	Guid	Gibt die GUID der Typversion zurück. ¹
ModifiedDate	DateTime	Gibt Datum und Uhrzeit zurück, zu denen die Typversion auf den Status "Committed" gesetzt wurde.
State	LibraryTypeVersionState	Gibt den Status der Version zurück: • InWork: Entspricht dem Status „In Bearbeitung“ oder „Im Test“, abhängig von dem zugeordneten Typ. • Committed: Entspricht dem Status „Freigegeben“.
TypeObject	LibraryType	Gibt den Typ zurück, zu dem diese Typversion gehört.
VersionNumber	Version	Gibt die Versionsnummer als dreistellige Versionskennung zurück, zum Beispiel „1.0.0“. ²

¹ Über dieses Attribut können Sie eine einzelne Typversion in einer Bibliothek finden.

² Über dieses Attribut können Sie eine einzelne Typversion in einer Zusammensetzung "LibraryTypeVersion" finden.

Alle Typversionen eines Typs enumerieren

Ändern Sie folgenden Programmcode:

```
//Enumerate the type versions of a type
public static void EnumerateVersionsInType(LibraryType libraryType)
{
    foreach (LibraryTypeVersion libraryTypeVersion in libraryType.Versions)
    {
        //...
    }
}
```

Attribute einer Typversion aufrufen

Ändern Sie folgenden Programmcode:

```
//Accessing the attributes of a type version
public static void InspectPropertiesOfVersion(LibraryTypeVersion libTypeVersion)
{
    string versionAuthor = libTypeVersion.Author;
    MultilingualText versionComment = libTypeVersion.Comment;
    Guid versionGUID = libTypeVersion.Guid; DateTime versionModifiedDate =
libTypeVersion.ModifiedDate;
    LibraryTypeVersionState versionStateLibrary = libTypeVersion.State;
    LibraryType versionParentObject = libTypeVersion.TypeObject;
    Version versionNumber = libTypeVersion.VersionNumber;
}
```

Instanz einer Typversion erstellen

Sie können eine neue Instanz einer Typversion erstellen. Die folgenden Objekte werden unterstützt:

- Bausteine (FB/FC)
- PLC-Anwenderdatentypen
- Bilder
- VB-Skripte

Eine Instanz einer Typversion kann aus der globalen Bibliothek oder der Projektbibliothek erzeugt werden. Wenn Sie eine Instanz einer Typversion aus einer globalen Bibliothek erstellen, wird die Typversion zunächst mit der Projektbibliothek synchronisiert.

Eine wiederherstellbare Ausnahme wird ausgelöst, wenn eine Instanz nicht im Ziel erstellt werden kann. Mögliche Ursachen:

- Die Bibliothekstypversion ist in Arbeit
- Eine Instanz der Bibliothekstypversion ist bereits im Zielgerät vorhanden

Ändern Sie folgenden Programmcode:

```
VBScriptLibraryTypeVersion scriptVersion = ...;
VBScriptComposition vbscripts = ...;

//Using the CreateFrom method to create an instance of the version in the VBScripts
composition
VBScript newScript = vbscripts.CreateFrom(scriptVersion);
```

Ändern Sie folgenden Programmcode:

```
ScreenLibraryTypeVersion screenVersion = ...;
ScreenComposition screens = ...;

//Using the CreateFrom method to create an instance of the version in the screens
composition
Screen newScreen = screens.CreateFrom(screenVersion);
```

Ändern Sie folgenden Programmcode:

```
CodeBlockLibraryTypeVersion blockVersion = ...;
PlcBlockComposition blocks = ...;

//Using the CreateFrom method to create an instance of the version in the blocks composition
PlcBlock newBlock = blocks.CreateFrom(blockVersion);
```

Ändern Sie folgenden Programmcode:

```
PlcTypeLibraryTypeVersion plcTypVersion=...;
PlcTypeComposition types=...;

//Using the CreateFrom method to create an instance of the version in the types composition
PlcType newType = types.CreateFrom(plcTypeVersion);
```

Verwendungen einer Typversion ermitteln

Die folgenden Verwendungen werden bei Typversionen unterschieden:

- Die Typversion verwendet andere Typversionen aus der Bibliothek.
Beispiel: Ein Anwenderdatentyp wird in einem Programmbaustein verwendet. Der Programmbaustein muss Zugriff auf den Anwenderdatentyp haben. Das bedeutet, dass der Programmbaustein vom Anwenderdatentyp abhängig ist.
Wenn Sie mit der Methode `GetDependencies()` auf das Abhängigkeitsattribut von `CodeBlockLibraryVersion` zugreifen, wird eine Liste von `LibraryTypeVersions` zurückgegeben.
- Der Typ wird von einer anderen Typversion in der Bibliothek verwendet.
Beispiel: Ein Anwenderdatentyp wird in einem Programmbaustein verwendet. Der Programmbaustein muss Zugriff auf den Anwenderdatentyp haben. Der Anwenderdatentyp hat den zugehörigen Programmbaustein. Der Programmbaustein ist vom Anwenderdatentyp abhängig.
Wenn Sie mit der Methode `GetDependents()` auf das Abhängigkeitsattribut von `PlcTypeLibraryTypeVersion` zugreifen, wird eine Liste von `LibraryTypeVersions` zurückgegeben.

Bei beiden Attributen wird eine Liste zurückgegeben, die Objekte des Typs `LibraryTypeVersion` enthält. Sind keine Verwendungen vorhanden, wird eine leere Liste zurückgegeben.

Hinweis

Wenn Sie diese Attribute bei Typversionen mit dem Status "InWork" verwenden, kann eine Ausnahme ausgelöst werden.

Ändern Sie folgenden Programmcode:

```
//Determine the uses of a type version in a library
public static void GetDependenciesAndDependentsOfAVersion(LibraryTypeVersion
libTypeVersion)
{
    IList<LibraryTypeVersion> versionDependents = libTypeVersion.Dependents();
    IList<LibraryTypeVersion> versionDependencies = libTypeVersion.Dependencies();
}
```

Programmcode

Um den Typ zu ermitteln, zu dem eine Typversion gehört, ändern Sie folgenden Programmcode:

```
public static void GetParentTypeOfVersion(LibraryTypeVersion libTypeVersion)
{
    LibraryType parentType = libTypeVersion.TypeObject;
}
```

Um die Masterkopien zu ermitteln, die Instanzen einer Typversion enthalten, ändern Sie folgenden Programmcode:

```
public static void GetMasterCopiesContainingInstances(LibraryTypeVersion libTypeVersion)
{
    MasterCopyAssociation masterCopies = libTypeVersion.MasterCopiesContainingInstances;
}
```

Um eine einzelne Typversion anhand ihrer Versionsnummer zu finden, ändern Sie folgenden Programmcode:

```
public static void FindVersionInLibrary(ILibrary library, Guid versionGUID)
{
    LibraryTypeVersion libTypeVersionByVersionNumber = library.FindVersion(versionGUID);
}
```

5.6.15 Zugriff auf Bausteine in Bibliotheken

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness den Versionsstand aus dem Bibliotheksobjekt abrufen, ohne das Objekt zu instanzieren und zu übersetzen.

Die folgende neue Export-Aktion wird im Engineering-Objekt `LibraryTypeVersion` zum Exportieren des Versionsinhalts bereitgestellt.

Die Aktion erstellt die exportierte XML-Datei auf dem mit `exportFileInfo` angegebenen Versionsstand.

```
void Export(''FileInfo'' exportFileInfo, ''ExportOptions'' exportOptions)
```

Es wird eine Ausnahme zurückgegeben:

- Wenn die Benutzerausnahme lautet "Die Versionsdaten können nicht exportiert werden, weil sie sich im Arbeitszustand befinden" und die zu exportierende Version den Zustand "Im Test" aufweist.
- Wenn eine allgemeine DataExchange-Benutzerausnahme, z B. "FileAlreadyExists" vorliegt, und die Meldung ausgegeben wird: "Der Export kann nicht durchgeführt werden, weil die Datei 'D:*.xml' bereits vorhanden ist."

Engineering-Objekt Bibliothekstypversion

Die folgenden Ergänzungen werden am Objekt `LibraryTypeVersion` engineering vorgenommen:

1. Die Export-Aktion wird an `LibraryTypeVersion` bereitgestellt.
2. Ein Navigator zum Navigieren zum durchsuchbaren Versionsinhalt.
 - Navigatorname: `ContentObject`
 - `ReadPublicationLevel` : `System`
 - Zusammenhangsname: `Engineering.Library.DefaultVersionContentObject`
 - Der Basisnavigator bietet eine Standard-Implementierung, um eine leere durchsuchbare Sammlung als Versionsinhalt bereitzustellen. Der Kunde kann diesen Navigator überschreiben und den Versionsinhalt bereitstellen.
3. "LibraryTypeName" und "LibraryTypeGuid" am Objekt `LibraryTypeVersion`.

Attribute	ReadPublicationLevel
LibraryTypeName	System
LibraryTypeGuid	System

Hinweis

Dies ist erforderlich, weil ohne diese Information nicht klar wäre, zu welchem Typ die exportierte Version (laut Versionsstand in der exportierten XML-Datei) gehört.

Exportierter Inhalt

Der folgende Inhalt wird in der exportierten Datei angezeigt:

- Exportierter Versionsstand der Bibliothek

Attribute	SimaticMLAAccess
Author	ReadWrite
Guid	ReadOnly
Modified Data	ReadOnly
VersionNumber	ReadWrite
LibraryTypeName	ReadOnly
Library TypeGuid	ReadOnly

Exportierte Navigatoren:

- Comment

Hinweis

Attribute mit schreibgeschütztem SimaticMLAAccess werden nur exportiert, wenn Sie die Exportoption "ExportOptions.WithReadOnly" wählen.

Programmcode

Ändern Sie folgenden Programmcode, um den Versionsstand des exportierten Inhalts zu exportieren:

```
Guid g = new Guid("35ad9996-d6d7-40c9-989f-0f0c7e21a7b2");  
//Block1  
var version = m_Project.ProjectLibrary.FindType(g).Versions[0];  
version.Export(new FileInfo(@"D:\ExportCodeBlock.xml"), ExportOptions.WithReadOnly);
```

Siehe auch

Projekt öffnen (Seite 131)

5.6.16 Auf Instanzen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek.
Siehe Auf globale Bibliotheken zugreifen (Seite 206).
- Sie haben Zugriff auf eine Gruppe für Typen.
Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 224).

Verwendung

Sie haben über die Schnittstelle TIA Portal Openness API Zugriff auf Instanzen von Typversionen.

Mit der Methode `FindInstances(IInstanceSearchScope searchScope)` können Sie alle Instanzen einer Typversion ermitteln.

Mit dem Parameter `searchScope` können Sie den Bereich des Projekts angeben, der durchsucht werden soll. Die folgenden Klassen implementieren die Schnittstelle `IInstanceSearchScope` und können zum Suchen nach Instanzen verwendet werden:

- `PlcSoftware`
- `HmiTarget`

Die Methode gibt eine Liste mit Objekten des Typs `LibraryTypeInfo` zurück. Sind keine Instanzen vorhanden, wird eine leere Liste zurückgegeben.

Hinweis

Instanzen von Bildbausteinen und HMI-Benutzerdatentypen sind stets mit der zugehörigen Typversion verknüpft.

Instanzen aller anderen Objekte wie Programmbausteine oder Bilder können mit einer Typversion verknüpft sein.

Instanzen einer Typversion enumerieren

Ändern Sie folgenden Programmcode:

```
//Enumerate the instances of a type version in the project
LibraryTypeVersion version = ...;
PlcSoftware plcSoftware = ...;

IInstanceSearchScope searchScope = plcSoftware as IInstanceSearchScope;

if(searchScope==null)
{
    //No search possible
}

IList<LibraryTypeInfo> instanceInfos = version.FindInstances(searchScope);
IEnumerable<IEngineeringObject> instances = instanceInfos.Select(instanceInfo =>
instanceInfo.LibraryTypeInfo);
```

Von einer Instanz zu dem damit verbundenen Versionsobjekt navigieren

Verwenden Sie das Attribut `LibraryTypeInfo` des Diensts `LibraryTypeInfo`, um von einer Instanz zu dem damit verbundenen Versionsobjekt zu navigieren.

Die folgenden Objekte bieten den Dienst `LibraryTypeInfo`:

- Bausteine FB
- Bausteine FC
- PLC-Anwenderdatentypen
- Bilder
- VB-Skripte

Wenn ein Instanzobjekt nicht mit einem Versionsobjekt verbunden ist, dann stellt es den Dienst "LibraryTypeInfo" nicht bereit.

```
FC fc = ...;
//Using LibraryTypeInfo service

LibraryTypeInfo instanceInfo = fc.GetService<LibraryTypeInfo>();
if(instanceInfo != null)
{
    LibraryTypeInfo connectedVersion = instanceInfo.LibraryTypeInfo;
    FC parentFc = instanceInfo.LibraryTypeInfo as FC; //parentFc == fc
}
```

5.6.17 Auf Kopiervorlagen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek.
Siehe Auf globale Bibliotheken zugreifen (Seite 206)
- Sie haben Zugriff auf eine Gruppe für Masterkopien.
Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 224)

Einleitung

Die Schnittstelle TIA Portal Openness API unterstützt den Zugriff auf Masterkopien in einer globalen Bibliothek und der Projektbibliothek:

- Masterkopien erstellen
- Masterkopien in Systemordnern und benutzerdefinierten Ordnern enumerieren
- Masterkopien umbenennen
- Informationen von Masterkopien abfragen
- Informationen von Objekten in einer Masterkopie abfragen

Attribut	Datentyp	Beschreibung
Author	String	Gibt den Namen des Autors zurück.
ContentDescriptions	MasterCopyContentDescriptionComposition	Gibt eine Beschreibung des Inhalts der Masterkopie zurück.
CreationDate	DateTime	Gibt das Erstellungsdatum zurück.
Name	String	Gibt den Namen der Masterkopie zurück.

Programmcode

Um alle Masterkopien im Systemordner einer Bibliothek zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateMasterCopiesInSystemFolder (MasterCopySystemFolder
masterCopySystemFolder)
{
    foreach (MasterCopy masterCopy in masterCopySystemFolder.MasterCopies)
    {
        //...
    }
}
```

5.6 Funktionen für Bibliotheken

Ändern Sie folgenden Programmcode, um mit der Methode "Find" auf eine einzelne Masterkopie zuzugreifen:

```
...
MasterCopySystemFolder systemFolder = projectLibrary.MasterCopyFolder;
MasterCopyComposition mastercopies = systemFolder.MasterCopies;
MasterCopy masterCopy = mastercopies.Find("Copy of ...");
...
```

Ändern Sie folgenden Programmcode, um Gruppen und Untergruppen von Masterkopien zu enumerieren:

```
private static void EnumerateFolder(MasterCopyFolder folder)
{
    EnumerateMasterCopies(folder.MasterCopies);
    foreach (MasterCopyUserFolder subFolder in folder.Folders)
    {
        EnumerateFolder(subFolder); // recursion
    }
}
private static void EnumerateMasterCopies(MasterCopyComposition masterCopies)
{
    foreach (MasterCopy masterCopy in masterCopies)
    {
        ...
    }
}
```

Ändern Sie den folgenden Programmcode, um mit der Methode "Find" auf einen MasterCopyUserFolder zuzugreifen:

```
...
MasterCopyUserFolderComposition userFolderComposition = ...
MasterCopyUserFolder userFolder = userFolderComposition.Find("Name of user folder");
...
```

Um eine Masterkopie umzubenennen, ändern Sie folgenden Programmcode:

```
//Setting the name attribute
var masterCopy = projectLibrary.MasterCopyFolder.MasterCopies.Find("SampleMasterCopyName");
masterCopy.Name = "NewMasterCopyName";

//Setting the name attribute dynamically
var masterCopy = projectLibrary.MasterCopyFolder.MasterCopies.Find("SampleMasterCopyName");
masterCopy.SetAttributes(new[] {new KeyValuePair<string,object>("Name",
"NewMasterCopyName")});
```

Informationen von Masterkopien abfragen

Um Informationen einer Masterkopie abzurufen, ändern Sie folgenden Programmcode:

```
public static void GetMasterCopyInformation(MasterCopy masterCopy)
{
    string author = masterCopy.Author;
    DateTime creationDate = masterCopy.CreationDate;
    string name = masterCopy.Name;
}
```

Informationen von Objekten in einer Masterkopie abfragen

Das Objekt MasterCopy enthält den Navigator ContentDescriptions, bei dem es sich um eine Zusammensetzung von MasterCopyContentDescriptions handelt.

Eine Masterkopie kann mehrere Objekte enthalten. Das Objekt MasterCopy enthält ContentDescriptions für jedes direkt in der Masterkopie enthaltene Objekt. Wenn die Masterkopie einen Ordner enthält, der ebenfalls einige Elemente enthält, enthält das Objekt MasterCopy nur eine ContentDescription des Ordners.

```
MasterCopy multiObjectMasterCopy = ...;

//Using ContentDescriptions
MasterCopyContentDescriptionComposition masterCopyContentDescriptions =
multiObjectMasterCopy.ContentDescriptions;
MasterCopyContentDescription contentDescription= masterCopyContentDescriptions.First();

string name = contentDescription.ContentName;
Type type = contentDescription.ContentType;
```

5.6.18 Masterkopie aus einem Projekt in Bibliothek erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Wenn eine Bibliothek eine Lesen-Schreiben-Bibliothek ist, können Sie eine Masterkopie aus einer IMasterCopySource-Quelle am Zielort erzeugen.

MasterCopy

```
MasterCopyComposition.Create(Siemens.Engineering.Library.MasterCopies.IMasterCopySource  
sourceObject);
```

Eine EngineeringException-Ausnahme wird in folgenden Fällen ausgelöst:

- Am Zielort besteht nur Lesezugriff
- Die Erzeugung der Masterkopie aus der Quelle wird vom System abgelehnt

Die folgenden Elemente sind als IMasterCopySources definiert:

- Device - HW
- DeviceItem - HW
- DeviceUserGroup - HW
- CodeBlock - SW
- DataBlock - SW
- PlcBlockUserGroup - SW
- PlcTag - SW
- PlcTagTable - SW
- PlcTagTableUserGroup - SW
- PlcType - SW
- PlcTypeUserGroup - SW
- PlcUnit - SW
- VBScript - HMI
- VBScriptUserFolder - HMI
- Screen - HMI
- ScreenTemplate - HMI
- ScreenTemplateUserFolder - HMI
- ScreenUserFolder - HMI
- Tag - HMI
- TagTable - HMI
- TagUserFolder - HMI

Programmcode

Verwenden Sie folgenden Programmcode:

```
// create a master copy from a code block in the project library
public static void Create(Project project, PlcSoftware plcSoftware)
{
    MasterCopySystemFolder masterCopyFolder = project.ProjectLibrary.MasterCopyFolder;
    CodeBlock block = plcSoftware.BlockGroup.Groups[0].Blocks.Find("Block_1") as CodeBlock;
    MasterCopy masterCopy = masterCopyFolder.MasterCopies.Create(block);
}
```

5.6.19 Ein Objekt aus einer Masterkopie erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Einleitung

Die Schnittstelle TIA Portal Openness API unterstützt die Verwendung von Masterkopien im Projekt. Sie können in der Zusammensetzung des Objekts aus einer Masterkopie in einer Projektbibliothek oder einer globalen Bibliothek mit Hilfe der Methode `CreateFrom` ein Objekt erstellen.

Der Ausgabetyt entspricht dem Ausgabetyt der jeweiligen Zusammensetzung.

Die Methode `CreateFrom` unterstützt nur Masterkopien, die einzelne Objekte enthalten. Wenn die Zusammensetzung, wo die Aktion aufgerufen wird, und die Quellmasterkopie inkompatibel sind (z. B. wenn die Quellmasterkopie eine PLC-Variablen-tabelle enthält und die Zusammensetzung die Zusammensetzung eines PLC-Bausteins ist), wird eine wiederherstellbare Ausnahme ausgelöst.

Die folgenden Zusammensetzungen werden unterstützt:

- `Siemens.Engineering.HW.DeviceComposition`
- `Siemens.Engineering.HW.DeviceItemComposition`
- `Siemens.Engineering.SW.Blocks.PlcBlockComposition`
- `Siemens.Engineering.SW.Tags.PlcTagTableComposition`
- `Siemens.Engineering.SW.Tags.PlcTagComposition`
- `Siemens.Engineering.SW.Types.PlcTypeComposition`
- `Siemens.Engineering.SW.TechnologicalObjects.TechnologicalInstanceDBComposition`

- Siemens.Engineering.SW.Tags.PlcUserConstantComposition
- Siemens.Engineering.Hmi.Tag.TagTableComposition
- Siemens.Engineering.Hmi.Tag.TagComposition
- Siemens.Engineering.Hmi.Screen.ScreenComposition
- Siemens.Engineering.Hmi.Screen.ScreenTemplateComposition
- Siemens.Engineering.Hmi.RuntimeScripting.VBScriptComposition
- Siemens.Engineering.HW.SubnetComposition
- Siemens.Engineering.HW.DeviceUserGroupComposition
- Siemens.Engineering.SW.Blocks.PlcBlockUserGroupComposition
- Siemens.Engineering.SW.ExternalSources.PlcExternalSourceUserGroupComposition
- Siemens.Engineering.SW.Tags.PlcTagTableUserGroupComposition
- Siemens.Engineering.SW.Types.PlcTypeUserGroupComposition

Programmcode: PLC-Baustein aus einer Masterkopie erstellen

Um einen PLC-Baustein aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie folgenden Programmcode:

```
var plcSoftware = ...;
MasterCopy copyOfPlcBlock = ...;
PlcBlock plcSoftware.BlockGroup.Blocks.CreateFrom(copyOfPlcBlock);
```

Programmcode: Ein Gerät aus einer Masterkopie erstellen

Um ein Gerät aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie folgenden Programmcode:

```
Project project = ...;
MasterCopy copyOfDevice = ...;
Device newDevice = project.Devices.CreateFrom(copyOfDevice);
```

Programmcode: Ein Geräteelement aus einer Masterkopie erstellen

Um ein Geräteelement aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie folgenden Programmcode:

```
Device device = ...;
MasterCopy copyOfDeviceItem = ...;
DeviceItem newDeviceItem = device.DeviceItems.CreateFrom(copyOfDeviceItem);
```


Programmcode: Ein Subnetz aus einer Masterkopie erstellen

Um ein Subnetz aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie folgenden Programmcode:

```
Project project = ...;  
MasterCopy copyOfSubnet = ...;  
Subnet newSubnet = project.Subnets.CreateFrom(copyOfSubnet);
```

Programmcode: Einen Geräteordner aus einer Masterkopie erstellen

Um einen Geräteordner aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie folgenden Programmcode:

```
Project project = ...;  
MasterCopy copyOfDeviceGroup = ...;  
DeviceGroup newDeviceGroup= project.DeviceGroups.CreateFrom(copyOfDeviceGroup);
```

Siehe auch

Auf Kopiervorlagen zugreifen (Seite 239)

5.6.20 Masterkopien kopieren**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Die Schnittstelle TIA Portal Openness API unterstützt das Kopieren von Masterkopien innerhalb einer Bibliothek und von einer Bibliothek zu einer anderen mit Hilfe der Aktion CreateFrom.

Die Aktion erstellt ein neues Objekt basierend auf der Quellmasterkopie und platziert es in der Zusammensetzung, wo die Aktion aufgerufen wurde. Die Aktion versucht, die neue Masterkopie mit dem gleichen Namen zu erstellen wie die Quellmasterkopie.

Wenn dieser Name nicht verfügbar ist, gibt das System der neuen Masterkopie einen neuen Namen. Danach gibt es die neue Masterkopie aus.

Wenn sich die Zusammensetzung, in der die Aktion "CreateFrom" aufgerufen wird, in einer schreibgeschützten globalen Bibliothek befindet, wird eine wiederherstellbare Ausnahme ausgelöst.

Programmcode

Ändern Sie folgenden Programmcode:

```
Project project = ...;
ProjectLibrary projectLibrary = project.ProjectLibrary;
MasterCopy copiedMasterCopy =
projectLibrary.MasterCopyFolder.MasterCopies.CreateFrom(sampleMasterCopy);
```

Siehe auch

Auf Kopiervorlagen zugreifen (Seite 239)

5.6.21 Ermitteln veralteter Typinstanzen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek.
Siehe Auf globale Bibliotheken zugreifen (Seite 206)
- Sie haben Zugriff auf einen Ordner für Typen.
Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 224).

Verwendung

Über die Schnittstelle TIA Portal OpennessAPI können Sie Typversionen ermitteln, die zu den Instanzen im geöffneten Projekt gehören. Dabei liefert die TIA Portal OpennessAPI pro Instanz einen der beiden folgenden Zustände:

- Die Instanz bezieht sich auf eine veraltete Typversion.
- Die Instanz bezieht sich auf die aktuelle Typversion.

Für die Versionsermittlung gelten die folgenden Regeln:

- Basis für die Versionsermittlung sind eine Bibliothek und das Projekt, das Sie über die Schnittstelle TIA Portal OpennessAPI öffnen möchten.
- Im Rahmen der Versionsermittlung werden keine Instanzen aktualisiert.

Signatur

Über die Methode `UpdateCheck` können Sie die Instanzen einer Typversion ermitteln:

```
UpdateCheck(Project project, UpdateCheckMode updateCheckMode)
```

Parameter	Funktion
Project	Gibt das Projekt an, in dem die Typversionen von Instanzen ermittelt werden.
UpdateCheckMode	Gibt die Versionen an, die ermittelt werden: - ReportOutOfDateOnly: Gibt nur den Status vom Typ „veraltet“ zurück. - ReportOutOfDateAndUpToDate: Gibt den Status der Typen „veraltet“ und „aktuell“ zurück.

Ergebnis

Bei der Versionsermittlung werden die Geräte des Projekts von oben nach unten abgefragt. Jedes Gerät wird darauf geprüft, ob seine Konfigurationsdaten eine Instanz einer Typversion aus der angegebenen Bibliothek enthalten. Die Methode `UpdateCheck` gibt das Ergebnis der Versionsprüfung in hierarchischer Reihenfolge zurück.

Die nachstehende Tabelle zeigt das Ergebnis einer Versionsprüfung mit dem Parameter `UpdateCheck.ReportOutOfDateAndUpToDate`:

Update check for: HMI_1		
	Update check for library element Screen_1 0.0.3	
	Out-of-date	
	\HMI_1\Screens	Screen_4 0.0.1
	\HMI_1\Screens	Screen_2 0.0.2
	Up-to-date	
	\HMI_1\Screens	Screen_1 0.0.3
	Update check for: HMI_2	
	Update check of library element Screen_4 0.0.3	
	Out-of-date	
	\Screens folder1	Screen_02 0.0.1
	\Screens folder1	Screen_07 0.0.2
	Up-to-date	
	\Screens folder1	Screen_05 0.0.3
	\Screens folder1	Screen_08 0.0.3

Programmcode

Es besteht kein Unterschied zwischen Projektbibliotheken und globalen Bibliotheken hinsichtlich der Handhabung der Aktualisierungsprüfung.

Um die Typversionen einer globalen Bibliothek oder Projektbibliothek für Instanzen im Projekt zu ermitteln, ändern Sie den folgenden Programmcode:

```
public static void UpdateCheckOfGlobalLibrary(Project project, ILibrary library)
{
    // check for out of date instances and report only out of date instances in the returned
    feedback
    UpdateCheckResult result = library.UpdateCheck(project,
    UpdateCheckMode.ReportOutOfDateOnly);

    //Alternatively, check for out of date instances and report both out of date and up to
    date instances in the returned feedback
    UpdateCheckResult alternateResult = library.UpdateCheck(project,
    UpdateCheckMode.ReportOutOfDateAndUpToDate);

    //Show result
    RecursivelyWriteMessages(result.Messages);

    // Alternatively, show result and access single message parts
    RecursivelyWriteMessageParts(result.Messages);
}
```

Um das Ergebnis der Versionsermittlung auszugeben und die Meldungen einzeln zu durchlaufen, ändern Sie folgenden Programmcode:

```
private static void RecursivelyWriteMessages (UpdateCheckResultMessageComposition
messages, string indent = "")
{
    indent += "\t";
    foreach (UpdateCheckResultMessage message in messages)
    {
        Console.WriteLine(indent + message.Description);
        RecursivelyWriteMessages(message.Messages, indent);
    }
}
```

Um im Ergebnis der Versionsermittlung auf einzelne Meldungsteile zuzugreifen, ändern Sie folgenden Programmcode:

```
private static void RecursivelyWriteMessageParts (UpdateCheckResultMessageComposition
messages, string indent= "")
{
    indent += "\t";
    foreach (UpdateCheckResultMessage message in messages)
    {
        Console.WriteLine(indent + "Full description: " + message.Description);
        foreach (KeyValuePair<string, string> messagePart in message.MessageParts)
        {
            // first level
            // part 1: device name
            // second level:
            // part 1: Name of the type in the global library
            // part 2: version of the type in the global library
            // third level:
            // part 1: title (either "Out-of-date" or "Up-to-date");
            // fourth level:
            // part 1: Path hierarchy to instance
            // part 2: Instance name in project
            // part 3: Version of the instance in the project
            Console.WriteLine(indent + "*Key: {0} Value:{1}", messagePart.Key,
messagePart.Value);
        }
        RecursivelyWriteMessageParts (message.Messages, indent);
    }
}
```

5.6.22 Projekt aktualisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek.
Siehe Auf globale Bibliotheken zugreifen (Seite 206).
- Sie haben Zugriff auf einen Ordner für Typen.
Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 224).

Verwendung

Über die Schnittstelle TIA Portal Openness API können Sie Instanzen eines ausgewählten Typs in einem Typordner eines Projekts aktualisieren.

Bei der Aktualisierung werden die im Projekt verwendeten Instanzen basierend auf der letzten freigegebenen Typversion aktualisiert. Wenn Sie die Instanzen in einer globalen Bibliothek aktualisieren, wird zunächst eine Synchronisierung durchgeführt.

Signatur

Zum Aktualisieren von Instanzen verwenden Sie die Methode `UpdateProject`.

Verwenden Sie für Klassen, die die Schnittstelle `LibraryTypes` implementieren, den folgenden Aufruf:

```
void UpdateProject(IUpdateProjectScope updateProjectScope)
```

Verwenden Sie für Klassen, die die Schnittstelle `ILibrary` implementieren, den folgenden Aufruf:

```
void UpdateProject(IEnumerable<ILibraryTypeOrFolderSelection>
selectedTypesOrFolders, IEnumerable <IUpdateProjectScope>
updateProjectScope)
```

Jeder Aufruf wird in die Protokolldatei im Projektverzeichnis eingetragen.

Parameter	Funktion
<code>IEnumerable<ILibraryTypeOrFolderSelection> selectedTypesOrFolders</code>	Gibt den Ordner oder zu synchronisierende Typen oder deren Instanzen im Projekt an, die aktualisiert werden sollen.
<code>IUpdateProjectScope updateProjectScope</code> <code>IEnumerable <IUpdateProjectScope> updateProjectScope</code>	Gibt die Objekte im Projekt an, in denen die Verwendungen der Instanzen aktualisiert werden sollen. Die folgenden Objekte werden unterstützt: • <code>PlcSoftware</code> • <code>HmiTarget</code>

Programmcode

Um Instanzen von ausgewählten Typen innerhalb eines Typordners zu aktualisieren, ändern Sie folgenden Programmcode:

```
private static void UpdateInstances(ILibrary myLibrary, LibraryTypeFolder
singleFolderContainingTypes, LibraryType singleType, PlcSoftware plcSoftware, HmiTarget
hmiTarget)
{
    //Update Instances of multiple types (subset of types and folders)
    IUpdateProjectScope[] updateProjectScopes =
    {
        plcSoftware as IUpdateProjectScope, hmiTarget as IUpdateProjectScope
    };
    myLibrary.UpdateProject(new ILibraryTypeOrFolderSelection[] {singleType,
singleFolderContainingTypes}, updateProjectScopes);
    //Update Instances of multiple types (all types in library)
    myLibrary.UpdateProject(new[] {myLibrary.TypeFolder}, updateProjectScopes);
}
```

5.6.23 Bibliothek aktualisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek.
Siehe Auf globale Bibliotheken zugreifen (Seite 206).
- Sie haben Zugriff auf einen Ordner für Typen.
Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 224).

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt die folgenden Aktualisierungen in der Projektbibliothek:

- Synchronisieren ausgewählter Typen zwischen Bibliotheken.

Bei der Synchronisierung wird die Ordnerstruktur nicht angepasst. Die zu aktualisierenden Typen werden über ihre GUID ermittelt und aktualisiert:

- Wenn ein Typ in einer Bibliothek eine Typversion enthält, die in der zu aktualisierenden Bibliothek fehlt, wird die Typversion kopiert.
- Der Vorgang wird abgebrochen und eine Exception ausgelöst, wenn ein Typ in einer Bibliothek eine Typversion mit unterschiedlicher GUID enthält.

Signatur

Zum Synchronisieren von Typversionen verwenden Sie die Methode `UpdateLibrary`.

Verwenden Sie für Klassen, die die Schnittstelle `LibraryTypes` implementieren, den folgenden Aufruf:

```
void UpdateLibrary(ILibrary targetLibrary)
```

Verwenden Sie für Klassen, die die Schnittstelle `ILibrary` implementieren, den folgenden Aufruf:

```
void UpdateLibrary(IEnumerable<LibraryTypeOrFolderSelection> selectedTypesOrFolders, ILibrary targetLibrary)
```

Parameter	Funktion
<code>IEnumerable<ILibraryTypeOrFolderSelection> selectedTypesOrFolders</code>	Gibt den Ordner oder zu synchronisierende Typen oder deren Instanzen im Projekt an, die aktualisiert werden sollen.
<code>ILibrary targetLibrary</code>	Gibt die Bibliothek an, deren Inhalt mit einer Bibliothek synchronisiert wird. Wenn Quellbibliothek und Zielbibliothek identisch sind, wird eine Ausnahme ausgelöst.

Programmcode

Um einen Typ aus einer Projektbibliothek mit einer globalen Bibliothek zu synchronisieren, ändern Sie den folgenden Programmcode:

```
sourceType.UpdateLibrary(projectLibrary);
```

Um ausgewählte Typen in einem Typordner zwischen einer globalen Bibliothek und der Projektbibliothek zu synchronisieren, ändern Sie folgenden Programmcode:

```
globalLibrary.UpdateLibrary(new[] {globalLibrary.TypeFolder}, projectLibrary);
```

5.6.24 Bibliothek bereinigen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit der TIA Portal Openness-Anwendung unbenutzte Versionen der angegebenen Typen in einer Bibliothek und alle Typen in einem bestimmten Ordner einer Bibliothek löschen.

Cleanup() steht für Projekt- und für globale Bibliotheken zur Verfügung; dazu müssen die Bibliotheken les- und schreibbar sein. Diese Funktion entspricht der Funktion "Bibliothek bereinigen" in der TIA Portal-Benutzeroberfläche.

Cleanup() führt seine Funktion entsprechend dem angegebenen Flag CleanupLibraryMode aus.

Eine Version wird abhängig vom jeweiligen Verhalten der beiden CleanupLibraryModes bereinigt:

CleanupLibraryMode	Beschreibung
CleanupLibraryMode.AllowTypeDeletion	Ermöglicht das Löschen aller Versionen eines Typs, der bereinigt werden soll. Sind alle Versionen eines Typs bereinigt, wird auch der Typ selbst bereinigt.
CleanupLibraryMode.PreserveHighestVersion	Die höchste Version eines Typs wird behalten

Programmcode

Ändern Sie folgenden Programmcode, um eine Teilmenge von Typen und Ordnern zu bereinigen:

```
...
ILibrary myLibrary = ...;
CleanupLibraryMode cleanupLibraryMode = ...;
ILibraryType typeA = ...; //from myLibrary
ILibraryType typeB = ...; //from myLibrary
LibraryTypeFolder singleFolderContainingTypes = ...; //from myLibrary
myLibrary.Cleanup(new[] {typeA, singleFolderContainingTypes}, cleanupLibraryMode);
...
```

Ändern Sie folgenden Programmcode, um alle Typen in einer Bibliothek zu bereinigen:

```
...
ILibrary myLibrary = ...;
CleanupLibraryMode cleanupLibraryMode = ...;
myLibrary.Cleanup(new [] {myLibrary.TypeFolder}, cleanupLibraryMode);
...
```

5.6.25 Projekt mit Projektbibliothek harmonisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Namen und Pfade von Typinstanzen mit den Typen in einer angegebenen Bibliothek harmonisieren. `HarmonizeProject()` ist sowohl für Projekt- als auch für globale Bibliotheken verfügbar. Diese Funktion entspricht der Harmonisierungsfunktion in der TIA Portal Benutzeroberfläche.

Sie können angeben, welche Geräte im Projekt harmonisiert werden sollen, und wählen, ob Namen, Pfade oder beides harmonisiert werden sollen. Wird die Funktion `HarmonizeProject` in einer globalen Bibliothek aufgerufen, wird die Projektbibliothek zuerst harmonisiert und dann das Projekt.

Die Harmonisierungsfunktion wird entsprechend den ausgewählten HarmonizeProjectOptions bearbeitet:

HarmonizeProjectOptions	Beschreibung
HarmonizePaths	Pfade der angegebenen Typen oder die Typen unter den angegebenen Ordnern harmonisieren. Die Instanzen im Projekt werden nach den Typhamen der Bibliothek umbenannt
HarmonizeProjectOptions.HarmonizeNames	Namen der angegebenen Typen oder die Typen unter den angegebenen Ordnern harmonisieren. Die Ordnerstruktur der Bibliothekstypen wird für die Projektinstanzen übernommen
HarmonizeProjectOptions.HarmonizeNames HarmonizeProjectOptions.HarmonizePaths	Namen und Pfade der ausgewählten Typen oder Typen unter einem ausgewählten Ordner harmonisieren
HarmonizeProjectOptions.None	Diese Option ist nicht zu verwenden, sonst wird eine Ausnahme ausgelöst

Einleitung

Ändern Sie den folgenden Programmcode, um Namen für eine Teilmenge von Typen und Ordnern zu harmonisieren:

```
...
ILibrary myLibrary = ...;
HarmonizeProjectOptions harmonizeProjectOptions = ...;
ILibraryType typeA = ...; //from myLibrary
ILibraryType typeB = ...; //from myLibrary
LibraryTypeFolder singleFolderContainingTypes = ...; //from myLibrary
IUpdateProjectScope harmonizeProjectScope = ..... //scope of device in project to be
harmonized
myLibrary.HarmonizeProject(new[]{typeA, singleFolderContainingTypes}, new[]
{harmonizeProjectScope, HarmonizeProjectOptions.HarmonizeNames});
...
```

Ändern Sie den folgenden Programmcode, um Pfade für eine Teilmenge von Typen und Ordnern zu harmonisieren:

```
...
ILibrary myLibrary = ...;
HarmonizeProjectOptions harmonizeProjectOptions = ...;
ILibraryType typeA = ...; //from myLibrary
ILibraryType typeB = ...; //from myLibrary
LibraryTypeFolder singleFolderContainingTypes = ...; //from myLibrary
IUpdateProjectScope harmonizeProjectScope = ..... //scope of device in project to be
harmonized
myLibrary.HarmonizeProject(new[]{typeA, singleFolderContainingTypes}, new[]
{harmonizeProjectScope, HarmonizeProjectOptions.HarmonizePaths});
...
```

Ändern Sie den folgenden Programmcode, um Namen und Pfade für eine Teilmenge von Typen und Ordnern zu harmonisieren:\

```
...
ILibrary myLibrary = ...;
HarmonizeProjectOptions harmonizeProjectOptions = ...;
ILibraryType typeA = ...; //from myLibrary
ILibraryType typeB = ...; //from myLibrary
LibraryTypeFolder singleFolderContainingTypes = ...; //from myLibrary
IUpdateProjectScope harmonizeProjectScope = ..... //scope of device in project to be
harmonized
myLibrary.HarmonizeProject(new[]{typeA, singleFolderContainingTypes}, new[]
{harmonizeProjectScope}, HarmonizeProjectOptions.HarmonizeNames |
HarmonizeProjectOptions.HarmonizePaths);
...
```

Ausnahmebehandlung

Die Ausnahmeszenarien sind nachstehend beschrieben:

NULL in sourceTypesAndFolder-Sammlung:

```
var projectLibrary = Project.ProjectLibrary;
var type1 = projectLibrary.Types.FindType("Type1"); // if this statement yields NULL
projectLibrary.HarmonizeProject(new[]{type1}, new[] {harmonizeProjectScope},
HarmonizeProjectOptions.HarmonizeNames); // throws EngineeringTargetInvocationException
```

NULL sourceTypesAndFolder-Sammlung:

```
projectLibrary.HarmonizeProject(null, new[] { harmonizeProjectScope },
HarmonizeProjectOptions.HarmonizeNames); // throws EngineeringTargetInvocationException
```

Leere sourceTypesAndFolder-Sammlung

```
projectLibrary.HarmonizeProject(new[] {}, new[] {harmonizeProjectScope},
HarmonizeProjectOptions.HarmonizeNames); // throws EngineeringTargetInvocationException
```

NULL in harmonizeProjectScope-Sammlung

```
var projectLibrary = Project.ProjectLibrary;
var type1 = projectLibrary.Types.FindType("Type1");
projectLibrary.HarmonizeProject(new[]{type1}, new[] { null },
HarmonizeProjectOptions.HarmonizeNames); // throws EngineeringTargetInvocationException
```

HarmonizeProjectOptions ist None

```
var projectLibrary = Project.ProjectLibrary;
var type1 = projectLibrary.Types.FindType("Type1");// if this statement yields NULL
projectLibrary.HarmonizeProject(new[]{type1}, new[] { harmonizeProjectScope },
HarmonizeProjectOptions.None); // throws EngineeringTargetInvocationException
```

sourceTypesAndFolder aus einer anderen Bibliothek

```
var type1 = projectLibrary.Types.FindType("Type1");
var type2 = globalLibrary.Types.FindType("Type2");
projectLibrary.HarmonizeProject(new[]{type1, type2}, new[] { harmonizeProjectScope },
HarmonizeProjectOptions.None); // throws EngineeringTargetInvocationException
```

Weitere Fälle für die EngineeringTargetInvocationException sind die folgenden:

- Das Projekt ist im Zustand ReadOnly
- Die Bibliothek ist im Zustand ReadOnly
- Die harmonizeScope sind aus unterschiedlichen Projekten
- Die harmonizeScope-Sammlung ist im Zustand ReadOnly

5.6.26 Struktur während der Bibliotheksaktualisierung aktualisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness das Projekt und die Struktur der globalen Bibliothek aktualisieren oder beibehalten, auch wenn Konflikte bestehen. Sie können auch mit TIA Portal Openness unbenutzte Versionen aus der Zielbibliothek

beibehalten oder löschen.

Die UpdateLibrary() wird mit der folgenden Signatur für die Projektbibliothek und die globale Bibliothek definiert:

```
public void UpdateLibrary(IEnumerable<ILibraryTypeOrFolderSelection>
sourceTypesAndFolders, ILibrary targetLibrary, ForceUpdateMode forceUpdateMode,
DeleteUnusedVersionsMode updateMode, StructureConflictResolutionMode
structureConflictResolutionMode);
```

Die UpdateLibrary() wird für Einzeltypen mit einer anderen Definition und folgender Signatur überladen:

```
public void UpdateLibrary(ILibrary targetLibrary, DeleteUnusedVersionsMode updateMode,  
StructureConflictResolutionMode structureConflictResolutionMode, ForceUpdateMode  
forceUpdateMode);
```

Die UpdateLibrary() akzeptiert folgende Parameterdefinition für die Ausführung verschiedener Vorgänge:

Parameter	Definition
DeleteUnusedVersionsMode	Dient zur Bestimmung, ob unbenutzte Versionen in der Zielbibliothek gelöscht/beibehalten werden sollen. DeleteUnusedVersionMode führt Vorgänge anhand des als Eingabe bereitgestellten ENUM-Werts aus: - DeleteUnusedVersionsMode.AutomaticallyDelete : Unbenutzte Versionen werden in der Zielbibliothek gelöscht, wenn in Zielbibliotheken unbenutzte Versionen vorhanden sind - DeleteUnusedVersionsMode.DoNotDelete: Unbenutzte Versionen werden in der Zielbibliothek nicht gelöscht, wenn in Zielbibliotheken unbenutzte Versionen vorhanden sind
StructureConflictResolutionMode	Dient zur Bestimmung, ob bei einem Strukturkonflikt die Struktur der Zielbibliothek aktualisiert, die vorhandene Struktur beibehalten oder der Vorgang abgebrochen werden soll. StructureConflictResolutionMode führt Vorgänge anhand des als Eingabe bereitgestellten ENUM-Werts aus: - StructureConflictResolutionMode.UpdateStructure: Die Struktur der Quellbibliothek wird anhand der Zielbibliothek aktualisiert, wenn es zwischen Quell- und Zielbibliothek strukturelle Unterschiede gibt - StructureConflictResolutionMode.RetainStructure: Die Struktur der Quellbibliothek wird in der Zielbibliothek beibehalten, wenn es zwischen Quell- und Zielbibliothek strukturelle Unterschiede gibt - StructureConflictResolutionMode.CancelIfStructureConflicts: Löst bei Auftreten eines Strukturkonflikts eine EngineeringTargetInvocationException aus und bricht den Vorgang ab.

Parameter	Definition
ForceUpdateMode	Dient zur Bestimmung, ob die aktualisierte Version in der Zielbibliothek als Standard-/neueste Version eingestellt werden soll. Der ForceUpdateMode führt Vorgänge anhand des als Eingabe bereitgestellten ENUM-Werts aus: - ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault: Die Standardversion in der Quellbibliothek wird im Ziel aktualisiert und die aktualisierte Version wird in der Zielbibliothek nur als Standardversion markiert, wenn die Versionsnummer in der Quelle höher als die Versionsnummer im Ziel ist - ForceUpdateMode.ForceSetAnyUpdatedVersionAsDefault: Die Standardversion in der Quellbibliothek wird unabhängig von der Versionsnummer im Ziel als Standard- oder neueste Version aktualisiert. - ForceUpdateMode.NoDefaultVersionChange : Dieser Enum-Wert sollte bei der Aktualisierung von der Projektbibliothek im Projekt verwendet werden, wenn die Option ForceUpdate nicht anwendbar ist

Programmcode: UpdateLibrary für einen Einzeltyp

Die UpdateLibrary API ist Teil der Siemens.Engineering.Library.Types.LibraryType-Schnittstelle und akzeptiert eine "Einfachauswahl" für Typen.

Ändern Sie den folgenden Programmcode, um UpdateLibrary für einen Einzeltyp aus der Projektbibliothek in der globalen Bibliothek auszuführen (unbenutzte Version löschen, vorhandene Struktur der Zielbibliothek aktualisieren und Aktualisierung der Version im Ziel standardmäßig unabhängig von der Versionsnummer erzwingen)

```
...
ILibrary myProjectLibrary = Project.ProjectLibrary;
LibraryTypeUserFolder singleFolderContainingTypes =
myProjectLibrary.TypeFolder.Folders.Find("folder1");
ILibraryType typeA = singleFolderContainingTypes.Types.Find("block1");
// block1 has lower version.
DeleteUnusedVersionsMode deleteUnusedVersionsMode =
DeleteUnusedVersionsMode.AutomaticallyDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.UpdateStructure;
ForceUpdateMode forceUpdateMode = ForceUpdateMode.ForceSetAnyUpdatedVersionAsDefault;
ILibrary globalLibrary = ...;
typeA.UpdateLibrary(globalLibrary, deleteUnusedVersionsMode,
structureConflictResolutionMode, forceUpdateMode);
...
```

5.6 Funktionen für Bibliotheken

Ändern Sie den folgenden Programmcode, um UpdateLibrary für einen Einzeltyp aus der Projektbibliothek in der globalen Bibliothek auszuführen (unbenutzte Version nicht löschen, vorhandene Struktur der Zielbibliothek beibehalten und keine Aktualisierung erzwingen):

```
...
ILibrary globalLibrary = ...;
LibraryTypeUserFolder singleFolderContainingTypes =
globalLibrary.TypeFolder.Folders.Find("folder1");
ILibraryType typeA = singleFolderContainingTypes.Types.Find("block1");// block1 has lower
version.
DeleteUnusedVersionsMode deleteUnusedVersionsMode = DeleteUnusedVersionsMode.DoNotDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.RetainStructure;
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
ILibrary myProjectLibrary = Project.ProjectLibrary;
typeA.UpdateLibrary(myProjectLibrary, deleteUnusedVersionsMode,
structureConflictResolutionMode, forceUpdateMode);
...
```

Ändern Sie den folgenden Programmcode, um UpdateLibrary für einen Einzeltyp aus der globalen Bibliothek in der Projektbibliothek auszuführen (unbenutzte Version nicht löschen, vorhandene Struktur der Zielbibliothek beibehalten und keine Aktualisierung erzwingen):

```
...
ILibrary globalLibrary = ...;
LibraryTypeUserFolder singleFolderContainingTypes =
globalLibrary.TypeFolder.Folders.Find("folder1");
ILibraryType typeA = singleFolderContainingTypes.Types.Find("block1");// block1 has lower
version.
DeleteUnusedVersionsMode deleteUnusedVersionsMode = DeleteUnusedVersionsMode.DoNotDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.RetainStructure;
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
ILibrary myProjectLibrary = Project.ProjectLibrary;
typeA.UpdateLibrary(myProjectLibrary, deleteUnusedVersionsMode,
structureConflictResolutionMode, forceUpdateMode);
...
```


Ändern Sie den folgenden Programmcode, um UpdateLibrary für einen Einzeltyp aus der globalen Bibliothek in der Projektbibliothek auszuführen (unbenutzte Version löschen, vorhandene Struktur der Zielbibliothek aktualisieren und keine Aktualisierung erzwingen):

```
...
ILibrary globalLibrary = ...;
LibraryTypeUserFolder singleFolderContainingTypes =
globalLibrary.TypeFolder.Folders.Find("folder1");
ILibraryType typeA = singleFolderContainingTypes.Types.Find("block1"); // block1 has lower
version.
DeleteUnusedVersionsMode deleteUnusedVersionsMode =
DeleteUnusedVersionsMode.AutomaticallyDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.UpdateStructure;
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
ILibrary myProjectLibrary = Project.ProjectLibrary;
typeA.UpdateLibrary(myProjectLibrary, deleteUnusedVersionsMode,
structureConflictResolutionMode);
...
```

Programmcode: UpdateLibrary für mehrere Typen

Die UpdateLibrary API ist Teil der Siemens.Engineering.Library.ILibrary-Schnittstelle und akzeptiert eine "Mehrfachauswahl" für Typen und Typordner.

Ändern Sie den folgenden Programmcode, um UpdateLibrary für einen Typ aus der Projektbibliothek in der globalen Bibliothek auszuführen:

```
...ILibrary myProjectLibrary = Project.ProjectLibrary;
LibraryTypeUserFolder singleFolderContainingTypes =
myProjectLibrary.TypeFolder.Folders.Find("folder1");
ILibraryType typeA = singleFolderContainingTypes.Types.Find("block1"); // block1 has lower
version.
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
DeleteUnusedVersionsMode deleteUnusedVersionsMode = DeleteUnusedVersionsMode.DoNotDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.UpdateStructure;
ILibrary globalLibrary;
myProjectLibrary.UpdateLibrary(new[] { typeA }, globalLibrary, forceUpdateMode,
deleteUnusedVersionsMode, structureConflictResolutionMode);
...
```

5.6 Funktionen für Bibliotheken

Ändern Sie den folgenden Programmcode, um UpdateLibrary für mehrere Typen aus der Projektbibliothek in der globalen Bibliothek auszuführen:

```
...
ILibrary myProjectLibrary = Project.ProjectLibrary;
LibraryTypeUserFolder singleFolderContainingTypes =
myProjectLibrary.TypeFolder.Folders.Find("folder1");
ILibraryType typeA = singleFolderContainingTypes.Types.Find("block1"); // block1 has lower
version.
ILibraryType typeB = ...;
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
DeleteUnusedVersionsMode deleteUnusedVersionsMode = DeleteUnusedVersionsMode.DoNotDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.UpdateStructure;
ILibrary globalLibrary; myProjectLibrary.UpdateLibrary(new[] { typeA, typeB },
globalLibrary, forceUpdateMode, deleteUnusedVersionsMode, structureConflictResolutionMode);
...
```

Ändern Sie den folgenden Programmcode, um UpdateLibrary für einen Ordner, der Typen enthält, aus der Projektbibliothek in der globalen Bibliothek auszuführen:

```
...
ILibrary myProjectLibrary = Project.ProjectLibrary;
LibraryTypeUserFolder singleFolderContainingTypes =
myProjectLibrary.TypeFolder.Folders.Find("folder1");
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
DeleteUnusedVersionsMode deleteUnusedVersionsMode = DeleteUnusedVersionsMode.DoNotDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.UpdateStructure;
ILibrary globalLibrary;
myProjectLibrary.UpdateLibrary(new[]
{ (ILibraryTypeOrFolderSelection)singleFolderContainingTypes }, globalLibrary,
forceUpdateMode, deleteUnusedVersionsMode, structureConflictResolutionMode);
...
```

Ändern Sie den folgenden Programmcode, um UpdateLibrary für den Systemordner "Types" aus der Projektbibliothek in der globalen Bibliothek auszuführen:

```
...
ILibrary myProjectLibrary = Project.ProjectLibrary;
LibraryTypeSystemFolder typeFolder = myProjectLibrary.TypeFolder;
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
DeleteUnusedVersionsMode deleteUnusedVersionsMode = DeleteUnusedVersionsMode.DoNotDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.RetainStructure;
ILibrary globalLibrary;
myProjectLibrary.UpdateLibrary(new[] { (ILibraryTypeOrFolderSelection)typeFolder },
globalLibrary, forceUpdateMode, deleteUnusedVersionsMode, structureConflictResolutionMode);
...
```

Ändern Sie den folgenden Programmcode, um UpdateLibrary für einen Typ aus der globalen Bibliothek in der Projektbibliothek auszuführen:

```
...
ILibrary globalLibrary = ...;
LibraryTypeUserFolder singleFolderContainingTypes =
globalLibrary.TypeFolder.Folders.Find("folder1");
ILibraryType typeA = singleFolderContainingTypes.Types.Find("block1"); // block1 has lower
version.
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
DeleteUnusedVersionsMode deleteUnusedVersionsMode = DeleteUnusedVersionsMode.DoNotDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.UpdateStructure;
ILibrary myProjectLibrary = Project.ProjectLibrary;
globalLibrary.UpdateLibrary(new[] { typeA }, myProjectLibrary, forceUpdateMode,
deleteUnusedVersionsMode, structureConflictResolutionMode);
...
```

Ändern Sie den folgenden Programmcode, um UpdateLibrary für einen Typ aus der globalen Bibliothek in der globalen Bibliothek auszuführen:

```
...
ILibrary globalLibrarySource = ...;
LibraryTypeUserFolder singleFolderContainingTypes =
globalLibrarySource.TypeFolder.Folders.Find("folder1");
ILibraryType typeA = singleFolderContainingTypes.Types.Find("block1"); // block1 has lower
version.
ForceUpdateMode forceUpdateMode = ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault;
DeleteUnusedVersionsMode deleteUnusedVersionsMode = DeleteUnusedVersionsMode.DoNotDelete;
StructureConflictResolutionMode structureConflictResolutionMode =
StructureConflictResolutionMode.UpdateStructure;
ILibrary globalLibraryTarget = ...;
globalLibrarySource.UpdateLibrary(new[] { typeA }, globalLibraryTarget, forceUpdateMode,
deleteUnusedVersionsMode, structureConflictResolutionMode);
...
```

Ausnahmebehandlung für Einzeltyp-Ausnahmebehandlung

Einige der Ausnahmeszenarien sind nachstehend beschrieben:

StructureConflictResolutionMode ist CancellIfStructureConflicts

```
var myProjectLibrary = Project.ProjectLibrary;
LibraryTypeSystemFolder typeFolder =
myProjectLibrary.TypeFolder.Folders.Find("folder1"); //from myProjectLibrary
var type1 = typeFolder.Types.FindType("Type1"); // if this structure is not available in
Global Library
type1.UpdateLibrary(globalLibrary, DeleteUnusedVersionsMode.DoNotDeleteUnusedVersions,
StructureConflictResolutionMode.CancelIfStructureConflicts,
ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault); // throws
EngineeringTargetInvocationException
```

Zielbibliothek ist NULL

```
var myProjectLibrary = Project.ProjectLibrary;
var type1 = projectLibrary.Types.FindType("Type1");
ILibrary globalLibrary = Null;
type1.UpdateLibrary(globalLibrary, DeleteUnusedVersionsMode.DoNotDeleteUnusedVersions,
StructureConflictResolutionMode.RetainStructure,
ForceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault); // throws
EngineeringTargetInvocationException
```

Bei falschen Enum-Werten wird eine `EngineeringTargetInvocationException` ausgelöst:

```
var myProjectLibrary = Project.ProjectLibrary;
LibraryTypeSystemFolder typeFolder = myProjectLibrary.TypeFolder.Folders.Find("folder1");//
from myProjectLibrary
var type1 = typeFolder.Types.FindType("Type1"); // if this structure is not available in
Global Library
type1 .UpdateLibrary(globalLibrary, (DeleteUnusedVersionsMode)42,
StructureConflictResolutionMode.RetainStructure, (ForceUpdateMode)22); // throws
EngineeringTargetInvocationException
```

Ausnahmebehandlung für Mehrfachtyp-Ausnahmebehandlung

Einige der Ausnahmeszenarien sind nachstehend beschrieben:

NULL in sourceTypesAndFolder-Sammlung

```
var myProjectLibrary = Project.ProjectLibrary;
var type1 = projectLibrary.Types.FindType("Type1"); // if this statement yields NULL
myProjectLibrary.UpdateLibrary(new[] { type1 }, globalLibrary,
forceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault,
DeleteUnusedVersionsMode.DoNotDeleteUnusedVersions,
StructureConflictResolutionMode.RetainStructure); // throws
EngineeringTargetInvocationException
```

`StructureConflictResolutionMode` ist `CancelIfStructureConflicts`

```
var myProjectLibrary = Project.ProjectLibrary;
LibraryTypeSystemFolder typeFolder =
myProjectLibrary.TypeFolder.Folders.Find("folder1"); //from myProjectLibrary
var type1 = typeFolder.Types.FindType("Type1"); // if this structure is not available in
Global Library
myProjectLibrary.UpdateLibrary(new[] { type1 }, globalLibrary,
forceUpdateMode.SetOnlyHigherUpdatedVersionAsDefault,
DeleteUnusedVersionsMode.DoNotDeleteUnusedVersions,
StructureConflictResolutionMode.CancelIfStructureConflicts); // throws
EngineeringTargetInvocationException
```

Bei falschen Enumerationswerten wird `EngineeringTargetInvocationException` zurückgegeben

```
var myProjectLibrary = Project.ProjectLibrary;  
LibraryTypeSystemFolder typeFolder =  
myProjectLibrary.TypeFolder.Folders.Find("folder1"); //from myProjectLibrary  
var type1 = typeFolder.Types.FindType("Type1"); // if this structure is not available in  
Global Library  
myProjectLibrary.UpdateLibrary(new[] { type1 }, globalLibrary, (ForceUpdateMode)22,  
(DeleteUnusedVersionsMode)42, StructureConflictResolutionMode.RetainStructure); // throws  
EngineeringTargetInvocationException
```

Hinweis

Weitere Fälle, in denen `EngineeringTargetInvocationException` ausgelöst wird, sind nachfolgend beschrieben:

- Wird versucht, eine Aktualisierung einer schreibgeschützten globalen Bibliothek durchzuführen, wird eine Ausnahme ausgelöst.
 - Ist einer der ausgewählten Typen oder Ordner Null oder nicht Teil der Quellbibliothek, wird eine Ausnahme ausgelöst.
-

Siehe auch

Projekt öffnen (Seite 131)

5.6.27 Bibliothekstyp aktualisieren

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Das Projekt ist geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Typen davon ausschließen, in der Projektbibliothek aktualisiert zu werden, oder Projekte davon ausschließen, in der globalen Bibliothek aktualisiert zu werden. Diese Funktion ist nützlich, wenn Sie ein Projekt oder eine Projektbibliothek mit Typen aktualisieren möchten, die eine Aktualisierung benötigen, und andere Typen auslassen möchten, die nicht aktualisiert werden müssen.

Die Funktion ist außerdem hilfreich, wenn sie verknüpfte Bibliotheken von verschiedenen Anwendern verteilen möchten oder ein Projekt oder eine Projektbibliothek mit vom Anbieter bereitgestellten neuen Funktionen aktualisieren möchten, statt zu warten, bis alle Anbieter die gesamte Bibliothek neu ausliefern.

Sie können das Attribut `SetForUpdate` nur mit nicht schreibgeschützten globalen Bibliothekstypen verwenden, bei denen ein Get-Vorgang mit allen Bibliothekstypen ausgeführt werden kann. Mit `SetForUpdate` wird für Typen in der Projektbibliothek immer 'True' zurückgegeben.

Die folgende API dient zur Markierung des Typs, der in der Projektbibliothek oder im Projekt aktualisiert werden muss.

Attributname	Dienst-Provider	Beschreibung
SetForUpdate	LibraryType	Dient zum Festlegen oder Abrufen der verteilbaren Eigenschaft eines Bibliothekstyps

Hinweis

Die `Siemens.Engineering.EngineeringNotSupportedException` wird für Sie ausgelöst, wenn Sie Folgendes versuchen:

- Einen Vorgang 'Set' mit Hilfe des Attributs `SetForUpdate` Attribut für Typen in der Projektbibliothek durchzuführen
- Einen Vorgang 'Set' mit Hilfe des Attributs `SetForUpdate` Attribut für schreibgeschützte Typen durchzuführen

Programmcode

Ändern Sie folgenden Programmcode so, dass der Bibliothekstyp für die Aktualisierung mit Hilfe des Attributs `SetForUpdate` markiert wird:

```
using Siemens.Engineering;
...
{
    TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
    string globalLibrrayPath =
@"C:\Proj\GlobalLibrary_SetForUpdate\GlobalLibrary_SetForUpdate.al18";
    FileInfo globalLibraryFileInfo = new FileInfo(globalLibrrayPath);
    var gl = portal.GlobalLibraries.Open(globalLibraryFileInfo, OpenMode.ReadWrite);
    var globalLibTypes = gl.TypeFolder.Types;
    //Get 'SetForUpdate' value for a specific library type
    Console.WriteLine(string.Format(":::GloblaLibrary:::SetforUpdateValue:::{0}",
globalLibTypes[0].SetForUpdate.ToString()));
    //Set 'SetForUpdate' value for a specific library type
    globalLibTypes[0].SetForUpdate = true;
}
```

5.6.28 Bibliotheksinhalte löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)
- Sie haben Zugriff auf die erforderliche Bibliothek.
Siehe Auf globale Bibliotheken zugreifen (Seite 206).
- Sie haben Zugriff auf einen Ordner für Typen.
Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 224).

Einleitung

Mit der TIA Portal Openness API-Schnittstelle können Sie die folgenden Inhalte in der Projektbibliothek löschen:

- Typen
- Typversion
- Benutzerdefinierte Ordner für Typen
- Masterkopien
- Benutzerdefinierte Ordner für Masterkopien

Hinweis

Löschen von Typen und Ordnern mit benutzerdefinierten Typen

Wenn Sie einen Typ oder einen Ordner mit benutzerdefinierten Typen löschen möchten, müssen die "Regeln zum Löschen von Versionen" erfüllt werden. Einen leeren Typordner können Sie jederzeit löschen.

Hinweis**Regeln zum Löschen von Versionen**

Sie können nur Versionen mit dem Status "Committed" löschen. Die folgenden Regeln gelten ebenfalls beim Löschen von Versionen:

- Wenn eine neue Version mit dem Status "In Arbeit" gerade aus einer Version mit dem Status "Committed" erstellt wurde, können Sie die Version mit dem Status "Committed" nur dann löschen, wenn die neue Version verworfen wird oder den Status "Committed" erhält.
 - Wenn ein Typ lediglich eine Version hat, wird auch der Typ gelöscht.
 - Wenn Version A von Version B eines anderen Typs abhängig ist, löschen Sie zunächst Version A und danach Version B.
 - Wenn es Instanzen von Version A gibt, können Sie Version A nur dann löschen, wenn auch die Instanzen gelöscht werden. Wenn eine Instanz auch in einer Masterkopie enthalten ist, wird die Masterkopie ebenfalls gelöscht.
-

Programmcode

Ändern Sie den folgenden Programmcode, um Typen oder Ordner mit benutzerdefinierten Typen zu löschen:

```
public static void DeleteMultipleTypesOrTypeUserFolders(ILibrary library)
{
    LibraryType t1 = library.TypeFolder.Types.Find("type1");
    LibraryType t2 = library.TypeFolder.Types.Find("type2");
    LibraryTypeUserFolder f1 = library.TypeFolder.Folders.Find("folder1");
    t1.Delete();
    t2.Delete();
    f1.Delete();
}
```

Um einen einzelnen Typ oder einen einzelnen Ordner mit benutzerdefinierten Typen zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteSingleTypeOrTypeUserFolder(ILibrary library)
{
    //Delete a single type
    LibraryType t1 = library.TypeFolder.Types.Find("type1");
    t1.Delete();

    //Delete a single folder
    LibraryTypeFolder parentFolder = library.TypeFolder;
    LibraryTypeUserFolder f1 = parentFolder.Folders.Find("folder1");
    f1.Delete();
}
```


Um eine Version zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteVersion(ILibrary library)
{
    LibraryType singleType = library.TypeFolder.Types.Find("type1");
    LibraryTypeVersion version1 = singleType.Versions.Find(new System.Version(1, 0, 0));
    version1.Delete();
}
```

Um eine Masterkopie oder einen Ordner mit benutzerdefinierten Masterkopien zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteMasterCopies(ILibrary library)
{
    // Delete master copy
    MasterCopy masterCopy = library.MasterCopyFolder.MasterCopies.Find("myMasterCopy");
    masterCopy.Delete();

    // Delete master copy user folder
    MasterCopyUserFolder masterUserFolder =
library.MasterCopyFolder.Folders.Find("myFolder");
    masterUserFolder.Delete();
}
```

Siehe auch

Auf Kopiervorlagen zugreifen (Seite 239)

5.6.29 Standardversion eines Typs festlegen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Öffnen eines Projekts (Seite 84)

Einleitung

Sie können mit TIA Portal Openness jede freigegebene Version als Standardversion dieses Typs in der Projektbibliothek und in der globalen Bibliothek festlegen.

Die Standardversion gilt als bevorzugte Version eines Typs.

Programmcode

```
ILibrary myLibrary = ....;
LibraryType typeA = ...; // from myLibrary
LibraryTypeVersion versionA = ...; // from typeA
versionA.setAsDefault(); // set the version as a default version of the type.
```

Ausnahmebehandlung

Eine Ausnahme "EngineeringTargetInvocationException" wird ausgegeben, wenn beim Aufruf der SetAsDefault API ein Fehler aufgetreten ist.

Ausnahmen finden sich in folgenden Szenarien:

- In der Projektbibliothek, wenn SetAsDefault() in der Test-/Bearbeitungsversion aufgerufen wird
- Wenn die globale Bibliothek im schreibgeschützten Modus ist

Siehe auch

Projekt öffnen (Seite 131)

5.6.30 Standardversion für einen Typ abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Standardversion eines Typs für die Projektbibliothek und die globale Bibliothek abrufen. Ferner können Sie mit TIA Portal Openness prüfen, ob eine Version die Standardversion ist oder nicht. Die Standardversion gilt als bevorzugte Version eines Typs.

Programmcode

```
//...
ILibrary myLibrary = ...;
LibraryType typeA = ...// from myLibrary
var typeADefaultVersion = typeA.Versions.First((version) => version.IsDefault);
//from default version , user can get details related to version like Author, Guid, etc.
var nameofType = typeADefaultVersion.LibraryTypeName;
var guid = typeADefaultVersion.Guid;
var versionNumber = typeADefaultVersion.VersionNumber;
var authoroftheVersion = typeADefaultVersion.Author;
var status = typeADefaultVersion.State;
//...
```

Siehe auch

Projekt öffnen (Seite 131)

5.6.31 Verbesserte Typkonsistenz

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Typen auf Inkonsistenzen prüfen. Die Typen können einen unterschiedlichen Konsistenzstatus haben. Der Typ ist konsistent, wenn der abhängige Typ die Standardversion des Abhängigkeitstyps verwendet. Der Typ ist inkonsistent, wenn der abhängige Typ nicht die Standardversion des Abhängigkeitstyps verwendet.

Der Konsistenzstatus des Ordners zeigt an, ob ein Typ unter der Struktur konsistent oder inkonsistent ist.

Die Status API gehört zur Klasse `Siemens.Engineering.Library.Types.LibraryType` und `Siemens.Engineering.Library.Types.LibraryTypeFolder` und hat den Typ `ConsistencyStatus`.

Damit ein Benutzer von TIA Portal Openness den Konsistenzstatus eines Typs/Ordners prüfen kann, wird das Attribut "Status" für folgende Elemente eingeführt:

- Systemordner
- Benutzerdefinierter Ordner
- Typ

Damit können Sie eine Liste inkonsistenter Typen erstellen, um bestimmte Operationen mit ihnen auszuführen:

```
var listOfInconsistentTypes = Folder.Types.Where(x => x.Status ==
ConsistencyStatus.DefaultVersionInconsistent);
```

Enum ConsistencyStatus unterstützt die folgenden Werte:

Wert	Beschreibung
DefaultVersionInConsistent	Abhängige Typen verwenden nicht die Standardversion von Abhängigkeitstypen
Consistent	Abhängige Typen verwenden die Standardversion von Abhängigkeitstypen

Programmcode

Ändern Sie folgenden Programmcode, um das Statusattribut für den Systemordner "Types" in der Projektbibliothek zu lesen:

```
...
ILibrary myProjectLibrary = Project.ProjectLibrary;
LibraryTypeSystemFolder typeFolder = myProjectLibrary.TypeFolder;
var systemTypeFolderConsistencyStatus = typeFolder.Status;
if(systemTypeFolderConsistencyStatus == ConsistencyStatus.DefaultVersionInConsistent)
{
    //Do something.
}
// List of InConsistentType
var listOfInCosistentTypes = typeFolder.Types.Where(x => x.Status ==
ConsistencyStatus.InConsistent);
foreach(var type in listOfInCosistentTypes)
{
    // From type, user can get details like Author, Guid etc
    var nameOfType = type.LibraryTypeName;
    var guid = type.Guid;
    var authorOfTheVersion = type.Author;
    var status = type.State;
}
...
```

Ändern Sie folgenden Programmcode, um das Statusattribut für "UserFolder" in der Projektbibliothek zu lesen:

```
...
ILibrary myProjectLibrary = Project.ProjectLibrary;
LibraryTypeUserFolder userFolder = myProjectLibrary.TypeFolder.Folders.Find("folder1");
var userTypeFolderConsistencyStatus = userFolder.Status;
if(userTypeFolderConsistencyStatus == ConsistencyStatus.DefaultVersionInConsistent)
{
    //Do something.
}
...
```

Ändern Sie folgenden Programmcode, um das Statusattribut für "Individual Type" in der Projektbibliothek zu lesen:

```
...
ILibrary myProjectLibrary = Project.ProjectLibrary;
LibraryTypeUserFolder userFolder = myProjectLibrary.TypeFolder.Folders.Find("folder1");
ILibraryType blockType = userFolder.Types.Find("block1");
var blockTypeConsistencyStatus = blockType.Status;
if(blockTypeConsistencyStatus == ConsistencyStatus.DefaultVersionInConsistent)
{
    //Do something.
}
...
```

Hinweis

Dieses Attribut kann auch in der globalen Bibliothek auf die gleiche Weise verwendet werden.

Ausnahmebehandlung

Ist einer der ausgewählten Typen oder Ordner Null oder nicht Teil der Quellbibliothek, wird eine Ausnahme `ArgumentNullException` ausgegeben.

```
var projectLibrary = Project.ProjectLibrary;
var type1 = projectLibrary.Types.FindType("Type1");
var status = typeFolder.Status; // throws ArgumentNullException
```

Siehe auch

Projekt öffnen (Seite 131)

5.6.32 PLC-Meldetextlisten und Masterkopien in Bibliotheken verwalten

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness PLC-Meldetextlisten als Masterkopie in eine Bibliothek kopieren bzw. PLC-Meldetextlisten aus Masterkopie-Ordern der Bibliothek löschen. Es können sowohl Projekt- als auch globale Bibliotheken verwendet werden.

Mit TIA Portal Openness werden außerdem folgende Funktionen unterstützt:

- Auf Anwender- und Systemtextlisten zugreifen
- Anwendertextlisten aktualisieren
- Anwendertextlisten löschen
- Textlistenamen und ListRange festlegen
- Auf Textlisteninformationen zugreifen, darunter ID, Name und Typ für Anwender bzw. System

Eigenschaft

Referenz von PlcAlarmTextListGroup (Navigatoren)

Benutzer-definierter Typ	Navigator	Datentyp	Beschreibung
PlcUnitBase PlcSoftware	PlcAlarmTextlist-Group	PlcAlarmTextlist-Group	Erste Textlistenebene. Hier werden EOM-Klassen für System- und Anwendertextlisten erstellt.

Navigatoren von PlcAlarmTextListGroup

Navigator	Datentyp
PlcAlarmSystemTextlists	PlcAlarmSystemTextlistComposition
PlcAlarmUserTextlists	PlcAlarmUserTextlistComposition

PlcAlarmTextlist

Die folgenden Eigenschaften werden bei PLC-Meldetextlisten unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
ID	Int32	ID der PLC-Meldetextlisten abrufen	Lesen
ListRange	PlcAlarmTextlistList-Range	Bereichstypen für PLC-Meldetextlisten abrufen. PlcAlarmTextlistListRange akzeptiert folgende Enum-Werte: • Dezimal: Dezimalwert eines Bereichstyps • Binär: Binärwert eines Bereichstyps • Bit: Bitwert eines Bereichstyps • Keiner: Kein Wert eines Bereichstyps	Lesen
Name	String	Namen der PLC-Meldetextlisten abrufen	Lesen

Navigator von PlcAlarmTextListGroup

Navigator	Datentyp
Comment	MultilingualText

PlcAlarmUserTextlist

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
ListRange	PlcAlarmTextlistList-Range	Bereichstypen für PLC-Meldetextlisten abrufen	Lesen/Schreiben
Name	String	Namen der PLC-Meldetextlisten abrufen	Lesen/Schreiben

Programmcode

```

TiaPortal tia = TiaPortal.GetProcesses().Select(x => x.Attach()).First();
Project project = tia.Projects.First();
Device device = project.Devices.First();
DeviceItem deviceItem = (DeviceItem)(from x in device.DeviceItems where
x.Name.Equals("PLC_1") select x).First();
SoftwareContainer softwareContainer =
((IEngineeringServiceProvider)deviceItem).GetService<SoftwareContainer>();
PlcSoftware plc = softwareContainer.Software as PlcSoftware;
var group = plc.PlcAlarmTextlistGroup;
var groupParent = group.Parent;
var systemTextlist = group.PlcAlarmSystemTextlists.First();
var listRangeType = systemTextlist.ListRange;
var systemTextlistParent = systemTextlist.Parent;
var userTextlist = group.PlcAlarmUserTextlists.First();
var userTextlistParent = userTextlist.Parent;
var listRange = systemTextlist.ListRange;
var comment = systemTextlist.Comment;
userTextlist.Name = "set text";
userTextlist.ListRange =
Siemens.Engineering.SW.Alarm.TextLists.PlcAlarmTextlistListRange.Bit;
userTextlist.Comment.Items.Find(project.LanguageSettings.ActiveLanguages.Find(new
CultureInfo("en-US", false))).Text = "EN comment";
userTextlist.Comment.Items.Find(project.LanguageSettings.ActiveLanguages.Find(new
CultureInfo("de-DE", false))).Text = "DE comment";
userTextlist.Delete();
MasterCopy masterCopy =
project.ProjectLibrary.MasterCopyFolder.MasterCopies.First<MasterCopy>(x => x.Name ==
"textlist_in_library");
PlcAlarmUserTextlists.CreateFrom(masterCopy);
masterCopy.Delete();
project.ProjectLibrary.MasterCopyFolder.MasterCopies.Create(userTextlist);

```

5.7 Funktionen für den Zugriff auf Geräte, Netzwerke und Verbindungen

5.7.1 Editor "Geräte & Netze" öffnen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können über die API-Schnittstelle den Editor „Geräte & Netze“ über eine der beiden folgenden Methoden öffnen:

- `ShowHwEditor(View.Topology oder View.Network oder View.Device)`: Öffnet den Editor „Geräte & Netze“ aus dem Projekt.
- `ShowInEditor(View.Topology oder View.Network oder View.Device)`: Zeigt das angegebene Gerät im Editor „Geräte & Netze“ an.

Über den Parameter `View` definieren Sie die Sicht, die beim Öffnen des Editors angezeigt wird:

- `View.Topology`
- `View.Network`
- `View.Device`

Programmcode

Um den Editor „Geräte & Netze“ zu öffnen, ändern Sie den folgenden Programmcode:

```
// Open topology view from project
private static void OpenEditorDevicesAndNetworksFromProject(Project project)
{
    project.ShowHwEditor(Siemens.Engineering.HW.View.Topology);
}
```


Um den Editor „Geräte & Netze“ für ein Gerät zu öffnen, ändern Sie folgenden Programmcode:

```
// Open topology view for given device
private static void OpenEditorDevicesAndNetworksFromDevice(Device device)
{
    device.ShowInEditor(Siemens.Engineering.HW.View.Topology);
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.7.2 PLC-Target und HMI-Target abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können ermitteln, ob eine Softwarebasis in der TIA Portal Openness API als PLC-Target (PlcSoftware) oder HMI-Target verwendet werden kann.

Programmcode: PLC-Target

Um festzustellen, ob ein Geräteelement als PLC-Target verwendet werden kann, verwenden Sie folgenden Programmcode:

```
// Returns PlcSoftware
private PlcSoftware GetPlcSoftware(Device device)
{
    DeviceItemComposition deviceItemComposition = device.DeviceItems;
    foreach (DeviceItem deviceItem in deviceItemComposition)
    {
        SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
        if (softwareContainer != null)
        {
            Software softwareBase = softwareContainer.Software;
            PlcSoftware plcSoftware = softwareBase as PlcSoftware;
            return plcSoftware;
        }
    }
    return null;
}
```

Programmcode: HMI-Target

Um festzustellen, ob ein Geräteelement als HMI-Target verwendet werden kann, ändern Sie folgenden Programmcode:

```
//Checks whether a device is of type hmitarget
private HmiTarget GetHmiTarget(Device device)
{
    DeviceItemComposition deviceItemComposition = device.DeviceItems;
    foreach (DeviceItem deviceItem in deviceItemComposition)
    {
        SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
        if (softwareContainer != null)
        {
            Software softwareBase = softwareContainer.Software;
            HmiTarget hmiTarget = softwareBase as HmiTarget;
            return hmiTarget;
        }
    }
    return null;
}
```

Siehe auch

Geräte enumerieren (Seite 353)

5.7.3 Auf Attribute eines Adressobjekts zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Für Schreibzugriff ist der PLC offline.

Verwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Attribute des Adressobjekts abrufen oder festlegen.

Ferner können Sie das aktuelle Prozessabbild einem OB zuweisen.

Auf die folgenden Attribute kann zugegriffen werden:

Attributname	Datentyp	Schreibbar	Zugriff	Beschreibung
IsynchronousMode	Bool	r/w	Dynamisches Attribut	Isochronen Modus aktivieren/deaktivieren
ProcessImage	Int32	r/w	Dynamisches Attribut	Partitionsnummer des Prozessabbilds festlegen/abrufen
InterruptObNumber	Int64	r/w	Dynamisches Attribut	Nummer des Alarm-Organisationsbausteins festlegen/abrufen (nur klassischer Controller)
StartAddress	Int32	r/w	Modelliertes Attribut	Neuen Wert für StartAddress festlegen/abrufen

Einschränkung

- Attribut `StartAddress`
 - Durch Festlegen von `StartAddress` kann implizit die `StartAddress` des gegenteiligen IO-Typs am Namensmodul geändert werden. Durch Ändern der Eingangsadresse wird die Ausgangsadresse geändert.
 - Der Schreibzugriff wird nicht für alle Geräte unterstützt.
 - Gepackte Adressen werden in TIA Portal Openness nicht unterstützt.
 - Durch Ändern der Adresse über TIA Portal Openness werden die zugewiesenen Variablen nicht neu verdrahtet.
- Attribut `InterruptObNumber`
 - Nur zugänglich in Einstellungen mit S7-300 oder S7-400 Controllern. Bei S7-400 Controllern wird der Schreibzugriff unterstützt.

Programmcode: Attribute eines Adressobjekts abrufen oder festlegen

Um auf den isochronen Modus eines Adressobjekts zuzugreifen, ändern Sie den folgenden Programmcode:

```
Address address= ...;

// read attribute
bool attributeValue = (bool)address.GetAttribute("IsochronousMode");

// write attribute
address.SetAttribute("IsochronousMode", true);
```

Um auf das Attribut `ProcessImage` eines Adressobjekts zuzugreifen, ändern Sie folgenden Programmcode:

```
Address address= ...;

// read attribute
int attributeValue = (int)address.GetAttribute("ProcessImage");

// write attribute
address.SetAttribute("ProcessImage", 7);
```

Um auf das Attribut `InterruptObNumber` eines Adressobjekts zuzugreifen, ändern Sie folgenden Programmcode:

```
Address address= ...;

// read attribute
long attributeValue = (long)address.GetAttribute("InterruptObNumber");

// write attribute
address.SetAttribute("InterruptObNumber", 42L);

//default value = 40
```

Um auf das Attribut `StartAddress` eines Adressobjekts zuzugreifen, ändern Sie folgenden Programmcode:

```
Address address= ...;

// read attribute
int attributeValue = (int)address.GetAttribute("StartAddress");

// write attribute
address.StartAddress = IntValueStartAddress;
```

Programmcode: Das aktuelle Prozessabbild einem OB zuweisen

Um das aktuelle Prozessabbild einem OB zuweisen, ändern Sie den folgenden Programmcode:

```
OB obX =...
Address address= ...;

// assign PIP 5 to obX

address.SetAttribute("ProcessImage", 5);

try
{
    address.AssignProcessImageToOrganizationBlock(obX);
} catch(RecoverableException e) {
    Console.WriteLine(e.Message);
}

// remove this PIP-OB assignment

try
{
    address.AssignProcessImageToOrganizationBlock(null);
} catch(RecoverableException e) {
    Console.WriteLine(e.Message);
}
```

5.7.4 Aufrufen eines Modulkanal**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Signalmodule wie analoge Eingangsmodule haben normalerweise mehrere Kanäle in einem einzelnen Modul. Normalerweise bieten Kanäle mehrere Funktionen gleichzeitig, z. B. kann ein analoges Eingangsmodul mit vier Kanälen vier Spannungswerte zur selben Zeit messen.

Zum Aufrufen aller Kanäle eines Moduls wird das Kanalattribut eines Geräteelements verwendet.

Programmcode: Attribute von Kanälen

Um auf Attribute eines Kanals zuzugreifen, ändern Sie folgenden Programmcode:

```
DeviceItem aiModule = ...
ChannelComposition channels = aiModule.Channels;
foreach (Channel channel in channels)
{
    ... // Work with the channel
}
```

Programmcode: Attribute ermitteln

Um die identifizierenden Attribute für die einzelnen Kanäle abzurufen, ändern Sie folgenden Programmcode:

```
Channel channel = ...
int channelNumber = channel.Number;
ChannelType type = channel.Type;
ChannelIoType ioType = channel.IoType;
```

Programmcode: Aufrufen eines einzelnen Kanals

Um die identifizierenden Attribute für den direkten Zugriff auf einen Kanal zu verwenden, ändern Sie folgenden Programmcode:

```
DeviceItem aiModule = ...
Channel channel = aiModule.Channels.Find(ChannelType.Analog, ChannelIoType.Input, 0);
... // Work with the channel
```

Kanaltypen

Wert	Beschreibung
ChannelType.None	Der Kanaltyp ist ungültig.
ChannelType.Analog	Der Kanaltyp ist analog.
ChannelType.Digital	Der Kanaltyp ist digital.
ChannelType.Technology	Der Kanaltyp ist technologisch.

Kanal-IO-Typen

Wert	Beschreibung
ChannelIoType.None	Der Kanal-IO-Typ ist ungültig.
ChannelIoType.Input	Ein Eingangskanal.
ChannelIoType.Output	Ein Ausgangskanal.
ChannelIoType.Complex	Komplexe IO-Typen, z. B. für technologische Kanäle.

5.8 Funktionen für Netzwerke

5.8.1 Subnetz anlegen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Subnetze können auf zwei verschiedene Arten angelegt werden:

- Erstellen eines Subnetzes, das mit einer Schnittstelle verbunden ist: Die Art der Schnittstelle, an der das Subnetz angelegt wird, bestimmt die Art des Subnetzes
- Erstellen eines Subnetzes, das nicht mit einer Schnittstelle verbunden ist.

Programmcode: Erstellen eines Subnetzes, das mit einem Knoten verbunden ist

Um ein Subnetz zu erstellen, ändern Sie folgenden Programmcode:

```
Node node = ...;  
Subnet subnet = node.CreateAndConnectToSubnet("NameOfSubnet");
```

Die folgenden Typkennungen werden verwendet:

- System:Subnet.Ethernet
- System:Subnet.Profibus
- System:Subnet.Mpi
- System:Subnet.Asi

Programmcode: Erstellen eines Subnetzes, das nicht mit einer Schnittstelle verbunden ist

Um ein Subnetz zu erstellen, ändern Sie folgenden Programmcode:

```
Project project = ...;  
SubnetComposition subnets = project.Subnets;  
  
Subnet newSubnet = subnets.Create("System:Subnet.Ethernet", "NewSubnet");
```

Die folgenden Typkennungen können verwendet werden:

- System:Subnet.Ethernet
- System:Subnet.Profibus
- System:Subnet.Mpi
- System:Subnet.Asi

5.8.2 Auf Subnetze zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Für mehrere netzwerkbezogene Merkmale, z. B. Zuordnung von Schnittstellen zu einem Subnetz, müssen Sie Subnetze im Projekt aufrufen. Subnetze sind typischerweise direkt auf Projektebene zusammengefasst.

Programmcode: Auf alle Subnetze eines Projekts zugreifen

Um auf alle Subnetze mit Ausnahme von internen Subnetzen eines Projekts zuzugreifen, ändern Sie folgenden Programmcode:

```
Project project = ...
foreach (Subnet net in project.Subnets)
{
    ... // Work with the subnet
}
```

Programmcode: Auf ein spezifisches Subnetz zugreifen

Um auf ein bestimmtes Subnetz anhand des Namens zuzugreifen, ändern Sie folgenden Programmcode:

```
Project project = ...
Subnet net = project.Subnets.Find("PROFIBUS_1");
{
    ... // Work with the subnet
}
```


Attribute eines Subnetzes

Ein Subnetz hat folgende Attribute:

```
Subnet net = ...;  
string name = net.Name;  
NetType type = net.NetType;
```

Netzwerktypen

Wert	Beschreibung
NetType.Unknown	Der Typ des Netzwerks ist unbekannt.
NetType.Ethernet	Der Typ des Netzwerks ist Ethernet.
NetType.Profibus	Der Typ des Netzwerks ist Profibus.
NetType.Mpi	Der Typ des Netzwerks ist MPI.
NetType.ProfibusIntegrated	Der Typ des Netzwerks ist integrierter Profibus.
NetType.Asi	Der Typ des Netzwerks ist ASi.
NetType.PcInternal	Der Typ des Netzwerks ist PC intern.
NetType.Ptp	Der Typ des Netzwerks ist PtP.
NetType.Link	Der Typ des Netzwerks ist Link.
NetType.Wan	Der Typ des Netzwerks ist Wide Area Network.

5.8.3 Auf interne Subnetze zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Wenn ein Geräteelement ein Subnetz bilden kann, verfügt es über die zusätzliche Funktionalität "Subnetzeigentümer". Um auf diese zusätzliche Funktionalität zuzugreifen, muss ein bestimmter Dienst des Geräteelements verwendet werden.

Programmcode: Subnetzeigentümerrolle abrufen

Um die Subnetzeigentümerrolle abzurufen, ändern Sie folgenden Programmcode:

```
SubnetOwner subnetOwner =  
(IEngineeringServiceProvider)deviceItem).GetService<SubnetOwner>();  
if (subnetOwner != null)  
{  
    // work with the role  
}
```

Programmcode: Attribute eines Subnetzeigentümers

Um auf die Subnetze eines Subnetzeigentümers zuzugreifen, verwenden Sie folgenden Programmcode:

```
foreach(Subnet subnet in subnetOwner.Subnets)  
{  
    Subnet interalSubnet = subnet;  
}
```

5.8.4 Typkennung von Subnetzen abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Das Attribut `TypeIdentifier` wird zur Identifizierung eines Subnetzes verwendet. `TypeIdentifier` ist eine Zeichenkette, die aus mehreren Teilen besteht: `<TypeIdentifierType>:<SystemIdentifier>`

Mögliche Werte für `TypeIdentifierType` sind:

- `System`

SystemIdentifier

Subnetztyp	Systemkennung
PROFIBUS	Subnet.Profibus
MPI	Subnet.Mpi
Industrial Ethernet	Subnet.Ethernet

Subnetztyp	Systemkennung
ASI	Subnet.Asi
Ptp	Subnet.Ptp
ProfibusIntegrated	Subnet.ProfibusIntegrated
PclInternal	Subnet.PclInternal
ProfidriveIntegrated	Subnet.ProfidriveIntegrated

Programmcode

Um die Typkennung für vom Benutzer verwaltbare und getrennt erstellbare Objekte für GSD abzurufen, ändern Sie den folgenden Programmcode:

```
Subnet subnet = ...;  
string typeIdentifier = subnet.TypeIdentifier;
```

5.8.5 Attribute eines Subnetzes aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Ein Subnetz bietet bestimmte obligatorische Attribute, die gelesen und/oder geschrieben werden können. Die Attribute sind nur verfügbar, wenn sie in der UI verfügbar sind. Der Schreibvorgang ist im Allgemeinen nur dann zulässig, wenn ein Attribut auch vom Benutzer in der UI geändert werden kann. Dies kann je nach Subnetztyp variieren. Der Benutzer kann DpCycleTime nur dann festlegen, wenn IsochronousMode wahr und DpCycleMinTimeAutoCalculation falsch ist.

Attribute von Subnetzen vom Typ ASI

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w	modelliert	Name des Subnetzes.
NetType	NetType	r	modelliert	Typ des Subnetzes
SubnetId	String	r	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.

Attribute von Subnetzen vom Typ Ethernet

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w	modelliert	Name des Subnetzes.
NetType	NetType	r	modelliert	Typ des Subnetzes
SubnetId	String	r/w	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.
DefaultSubnet	Bool	r/w	dynamisch	Wahr, wenn das Subnetz ein Standardsubnetz ist. In einem Projekt ist höchstens ein Standardsubnetz vorhanden.

Attribute von Subnetzen vom Typ MPI

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w	modelliert	Name des Subnetzes.
NetType	NetType	r	modelliert	Typ des Subnetzes
SubnetId	String	r/w	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.
HighestAddress	Int	r/w	dynamisch	Höchste MPI-Adresse am Subnetz.
TransmissionSpeed	BaudRate	r/w	dynamisch	Wahr, wenn das Subnetz ein Standardsubnetz ist. In einem Projekt ist höchstens ein Standardsubnetz vorhanden.

Attribute von Subnetzen vom Typ PC internal

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r	modelliert	Name des Subnetzes.
NetType	NetType	r	modelliert	Typ des Subnetzes
SubnetId	String	r	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.

Attribute von Subnetzen vom Typ PROFIBUS

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w	modelliert	Name des Subnetzes.
NetType	NetType	r	modelliert	Typ des Subnetzes
SubnetId	String	r/w	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.
HighestAddress	Int	r/w	dynamisch	Höchste PROFIBUS-Adresse am Subnetz.
TransmissionSpeed	BaudRate	r/w	dynamisch	Wahr, wenn das Subnetz ein Standardsubnetz ist. In einem Projekt ist höchstens ein Standardsubnetz vorhanden.
BusProfile	Busprofil	r/w	dynamisch	Das PROFIBUS-Profil.
PbCableConfiguration	Bool	r/w	dynamisch	Wahr, um zusätzliche PROFIBUS-Netzwerkeinstellungen zu aktivieren
PbRepeaterCount	Int	r/w	dynamisch	Anzahl der Repeater für Kupferleitung
PbCopperCableLength	double	r/w	dynamisch	Die Länge der Kupferleitung
PbOpticalComponentCount	Int	r/w	dynamisch	Anzahl der OLMs und OBTs von LWL-Leitung.
PbOpticalCableLength	double	r/w	dynamisch	Die Länge der LWL-Leitung für das PROFIBUS-Netzwerk in km.
PbOpticalRing	Bool	r/w	dynamisch	Wahr, wenn Busparameter für einen optischen Ring übernommen werden
PbOlmP12	Bool	r/w	dynamisch	Wahr, wenn OLM/P12 für die Berechnung des Busparameters aktiviert ist.
PbOlmG12	Bool	r/w	dynamisch	Wahr, wenn OLM/G12 für die Berechnung des Busparameters aktiviert ist.

5.8 Funktionen für Netzwerke

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
PbOlmG12Eec	Bool	r/w	dynamisch	Wahr, wenn OLM/G12-EEC für die Berechnung des Busparameters aktiviert ist.
PbOlmG121300	Bool	r/w	dynamisch	Wahr, wenn OLM/G12-1300 für die Berechnung des Busparameters aktiviert ist.
PbAdditionalNetworkDevices	Bool	r/w	dynamisch	Wahr, wenn zusätzliche Busgeräte, die im Projekt nicht vorhanden sind, bei der Berechnung der Buszeiten berücksichtigt werden.
PbAdditionalDpMaster	Int	r/w	dynamisch	Anzahl der nicht konfigurierten DP-Master.
PbTotalDpMaster	Int	r	dynamisch	Gesamtanzahl der DP-Master
PbAdditionalPassiveDevice	Int	r/w	dynamisch	Anzahl der nicht konfigurierten DP-Slaves oder passiven Geräte.
PbTotalPassiveDevice	Int	r	dynamisch	Gesamtanzahl der DP-Slaves oder passiven Geräte.
PbAdditionalActiveDevice	Int	r/w	dynamisch	Anzahl der nicht konfigurierten aktiven Geräte mit FDL/FMS/S/-Kommunikationslast.
PbTotalActiveDevice	Int	r	dynamisch	Gesamtanzahl der aktiven Geräte mit FDL/FMS/S/-Kommunikationslast.
PbAdditionalCommunicationLoad	CommunicationLoad	r/w	dynamisch	Grobe Quantifizierung der Kommunikationslast
PbDirectDataExchange	Bool	r/w	dynamisch	Optimierung für direkten Datenaustausch.
PbMinimizeTslotForSlaveFailure	Bool	r/w	dynamisch	Minimierung für Zeitzuweisung für Slave-Fehler.
PbOptimizeCableConfiguration	Bool	r/w	dynamisch	Optimierung der Leitungskonfiguration.
PbCyclicDistribution	Bool	r/w	dynamisch	Wahr, wenn zyklische Verteilung der Busparameter aktiviert ist.
PbTslotInit	Int	r/w	dynamisch	Standardwert von Tslot.
PbTslot	Int	r	dynamisch	Warte-auf-Empfangszeit (slot time)
PbMinTsdr	Int	r/w	dynamisch	Minimale Dauer für die Protokollverarbeitung
PbMaxTsdr	Int	r/w	dynamisch	Maximale Dauer für die Protokollverarbeitung
PbTid1	Int	r	dynamisch	Leerlaufzeit 1
PbTid2	Int	r	dynamisch	Leerlaufzeit 2
PbTrdy	Int	r	dynamisch	Bereitschaftszeit
PbTset	Int	r/w	dynamisch	Einrichtungszeit
PbTqui	Int	r/w	dynamisch	Modulator-Ausklingszeit

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
PbTtr	int64	r/w	dynamisch	Der Ttr-Wert in t_Bit.
PbTtrTypical	int64	r	dynamisch	Mittlere Antwortzeit am Bus
PbWatchdog	int64	r/w	dynamisch	Ansprechüberwachung
PbGapFactor	Int	r/w	dynamisch	GAP-Aktualisierungsfaktor
PbRetryLimit	Int	r/w	dynamisch	Maximale Anzahl an Wiederholungen
IsochronousMode	Bool	r/w	dynamisch	Wahr, wenn konstante Buszykluszeit aktiviert ist.
PbAdditionalPassivDeviceForIsochronousMode	Int	r/w	dynamisch	Anzahl der zusätzlichen OPs/PGs/TDs usw., die in dieser Netzwerksicht nicht konfiguriert sind.
PbTotalPassivDeviceForIsochronousMode	Int	r	dynamisch	Summe der konfigurierten und nicht konfigurierten Geräte, z. B. OPs/PGs/TDs usw.
DpCycleMinTimeAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der kürzesten DP-Zykluszeit aktiviert sind.
DpCycleTime	double	r/w	dynamisch	Die DP-Zykluszeit.
IsochronousTiToAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der Werte IsochronousTi und IsochronousTo aktiviert sind.
IsochronousTi	double	r/w	dynamisch	Time Ti (Zeit zum Lesen von Prozesswerten)
IsochronousTo	double	r/w	dynamisch	Time To (Zeit zum Ausgeben von Prozesswerten)

Attribute von Subnetzen vom Typ PROFIBUS Integrated

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w	modelliert	Name des Subnetzes.
NetType	NetType	r	modelliert	Typ des Subnetzes
SubnetId	String	r/w	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.
IsochronousMode	Bool	r	dynamisch	Konstante Buszykluszeit aktiviert.
DpCycleMinTimeAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der kürzesten DP-Zykluszeit aktiviert sind.
DpCycleTime	double	r/w	dynamisch	Die DP-Zykluszeit.

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
IsochronousTiToAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der Werte IsochronousTi und IsochronousTo aktiviert sind.
IsochronousTi	double	r/w	dynamisch	Time Ti (Zeit zum Lesen von Prozesswerten)
IsochronousTo	double	r/w	dynamisch	Time To (Zeit zum Ausgeben von Prozesswerten)

Attribute von Subnetzen vom Typ PROFIdrive Integrated

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w	dynamisch	Name des Subnetzes
NetType	NetType	r	modelliert	Typ des Subnetzes
IsochronousMode	Bool	r	dynamisch	Konstante Buszykluszeit aktiviert
SourceCycleTime*	Int32	r/w	dynamisch	Die Zykluszeit der Busquelle
DpCycleTime	double	r/w	dynamisch	Die DP-Zykluszeit
IsochronousTiToAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der Werte Ti (Lesen von Prozesswerten) und To (Ausgeben von Prozesswerten) aktiviert sind.
IsochronousTi	double	r/w	dynamisch	Time Ti (Zeit zum Lesen von Prozesswerten)
IsochronousTo	double	r/w	dynamisch	Time To (Zeit zum Ausgeben von Prozesswerten)

*SourceCycleTime unterstützt die folgenden Enum-Werte:

Enum-Name	Wert
Manual	0
AutomaticMinimum	1
LocalSendClock	2
ProfinetSendClock	3

Programmcode

Um die Attribute eines Subnetzes abzurufen oder festzulegen, ändern Sie folgenden Programmcode:

```
SubnetOwner subnetOwner =
    ((IEngineeringServiceProvider)deviceItem).GetService<SubnetOwner>();
Subnet subnet = subnetOwner.Subnets[0];
string nameValue = subnet.Name;
NetType nodeType = (NetType)subnet.NetType;
string subnetId = ((IEngineeringObject)subnet).GetAttribute("SubnetId");
subnet.Name = "NewName";
subnet.SetAttribute("Name", "NewName");
bool isDefaultSubnet = ((IEngineeringObject)subnet).GetAttribute("DefaultSubnet");
```


Baud-Raten

Wert	Beschreibung
BaudRate.None	Die Baud-Rate ist unbekannt.
BaudRate.Baud9600	9,6 kBaud
BaudRate.Baud19200	19,2 kBaud
BaudRate.Baud45450	45,45 kBaud
BaudRate.Baud93700	93,75 kBaud
BaudRate.Baud187500	187,5 kBaud
BaudRate.Baud500000	500 kBaud
BaudRate.Baud1500000	1,5 MBaud
BaudRate.Baud3000000	3 MBaud
BaudRate.Baud6000000	6 MBaud
BaudRate.Baud12000000	12 MBaud

Busprofile

Wert	Beschreibung
BusProfile.None	Das Busprofil ist unbekannt.
BusProfile.DP	Der Typ des Netzwerks ist DP.
BusProfile.Standard	Der Typ des Netzwerks ist Standard.
BusProfile.Universal	Der Typ des Netzwerks ist Universal.
BusProfile.UserDefined	Der Typ des Netzwerks ist benutzerdefiniert.

Kommunikationslast

Wert	Beschreibung
CommunicationLoad.None	Keine gültige Kommunikationslast.
CommunicationLoad.Low	Typisch für DP, keine größere Datenkommunikation außer DP.
CommunicationLoad.Medium	Typisch für Mischbetrieb von DP und anderen Kommunikationsdiensten (z. B. S7-Kommunikation), wenn DP hohe Zeitanforderungen hat und bei mittlerem azyklischen Kommunikationsaufkommen.
CommunicationLoad.High	Für Mischbetrieb von DP und anderen Kommunikationsdiensten (z. B. S7-Kommunikation), wenn DP geringe Zeitanforderungen hat und bei hohem azyklischen Kommunikationsaufkommen.

5.8.6 Globales Subnetz löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um ein globales Subnetz innerhalb eines Projekts zu löschen, ändern Sie folgenden Programmcode:

```
Project project = ...;
SubnetComposition subnets = project.Subnets;
subnets.First();
// delete subnet
Subnet subnetToDelete = ...;
subnetToDelete.Delete();
```

5.8.7 Alle Beteiligten eines Subnetzes enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Enumeration aller Beteiligten eines Subnetzes.

Programmcode

Um DP-Mastersysteme aus dem Subnetz zu enumerieren, ändern Sie folgenden Programmcode:

```
Subnet subnet = ...;
foreach (Node node in subnet.Nodes)
{
    // work with the node
}
```

5.8.8 IO-Systeme eines Subnetzes enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Die Enumeration von IoSystem stellt alle IO-Systeme, die sich auf dem Subnetz befinden, bereit. Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt.

Programmcode

Um DP-Mastersysteme aus dem Subnetz zu enumerieren, ändern Sie folgenden Programmcode:

```
Subnet subnet = ...;
foreach (IoSystem ioSystem in subnet.IoSystems)
{
    // work with the io system
}
```

5.8.9 Auf Knoten zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die Rollenschnittstelle aggregiert Knoten: Um auf Attribute zuzugreifen, die mit der Adress- und Subnetzzuordnung einer Schnittstelle zusammenhängen.

Der Name eines Knotens wird in den Attributen einer Schnittstelle im TIA Portal angezeigt. Die Knotenkennung ist eine einzigartige Kennung für jeden auf einer Schnittstelle erfassten Knoten. Der Wert kann nur über TIA Portal Openness angezeigt werden.

Programmcode

Um auf alle Knoten einer Schnittstelle zuzugreifen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (Node node in itf.Nodes)
{
    ... // Work with the node
}
```

Die meisten Schnittstellen haben nur einen einzigen Knoten, daher wird normalerweise der erste Knoten verwendet.

```
NetworkInterface itf = ...
Node node = itf.Nodes.First();
{
    ... // Work with the node
}
```

Knoten stellen ihre Namen und Typen und Knotenkennungen als Attribute zur Verfügung:

```
Node node = ...
string name = node.Name;
NetType type = node.NodeType;
string id = node.NodeId;
```

5.8.10 Attribute eines Knotens aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Ein Geräteelement bietet bestimmte obligatorische Attribute, die gelesen und/oder geschrieben werden können. Die Attribute sind nur verfügbar, wenn sie in der UI verfügbar sind. Der Schreibvorgang ist im Allgemeinen nur dann zulässig, wenn ein Attribut auch vom Benutzer in der UI geändert werden kann. Dies kann je nach Typ des Geräteelements variieren. Der Benutzer kann RouterAddress nur dann festlegen, wenn RouterUsed wahr ist. Wenn der Benutzer die SubnetMask am IO-Controller ändert, dann wird auch die Subnetzmaske auf allen IO-Devices auf denselben Wert geändert.

Attribute eines Knotens vom Typ ASI

Attribute	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r	modelliert	Name des Knotens.
Nodeld	String	r	modelliert	ID des Knotens.
NodeType	NetType	r	modelliert	Ein Knoten ruft seinen Typen vom Subnetz ab.
Address	String	r/w	dynamisch	Weitere Attribute für AS-i-Slaves.

Attribute eines Knotens vom Typ Ethernet

Attribute	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r	modelliert	Name des Knotens.
Nodeld	String	r	modelliert	ID des Knotens.
NodeType	NetType	r/w manchmal r/o	modelliert	Ein Knoten ruft seinen Typen vom Subnetz ab.
UseIsoProtocol	Bool	r/w	dynamisch	True, wenn ISO-Protokoll verwendet werden muss
MacAddress	String	r/w	dynamisch	z. B. 01-80-C2-00-00-00
UseIpProtocol	Bool	r/w	dynamisch	Dieser Wert kann auch dann gelesen werden, wenn er an der entsprechenden TIA UI-Steuerung nicht sichtbar ist.
IpProtocolSelection	Enum	r/w	dynamisch	-
Address	String	r/w	dynamisch	nur IPv4 und nicht IPv6 wird unterstützt
SubnetMask	String	r/w	dynamisch	-
UseRouter	Bool	r/w	dynamisch	-
RouterAddress	String	r/w	dynamisch	-
DhcpClientId	String	r/w	dynamisch	-
PnDeviceNameSetDirectly	Bool	r/w	dynamisch	PROFINET-Gerätename wird direkt am Gerät festgelegt. Nicht für jedes Gerät verfügbar.
PnDeviceNameAutoGeneration	Bool	r/w	dynamisch	PROFINET-Gerätename wird automatisch erstellt.

Attribute	Datentyp	Schreibbar	Zugriff	Beschreibung
PnDeviceName	String	r/w	dynamisch	Eindeutiger Name im Subnetz.
PnDeviceNameConverted	String	r	dynamisch	Gerätename für systeminterne Nutzung umgewandelt.

Attribute eines Knotens vom Typ MPI

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r	modelliert	Name des Knotens.
NodeId	String	r	modelliert	ID des Knotens.
NodeType	NetType	r	modelliert	Ein Knoten ruft seinen Typen vom Subnetz ab.
Address	String	r/w	dynamisch	-

Attribute eines Knotens vom Typ PC internal

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r	modelliert	Name des Knotens.
NodeId	String	r	modelliert	ID des Knotens.
NodeType	NetType	r	modelliert	Ein Knoten ruft seinen Typen vom Subnetz ab.

Attribute eines Knotens vom Typ PROFIBUS

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r	modelliert	Name des Knotens.
NodeId	String	r	modelliert	ID des Knotens.
NodeType	NetType	r	modelliert	Ein Knoten ruft seinen Typen vom Subnetz ab.
Address	String	r/w	dynamisch	Netzwerkadresse des Knotens. Der Typ der Adresse ist abhängig vom Knotentyp (z. B. IP-Adresse für PROFINET-Knoten, PROFIBUS-Adresse für PROFIBUS-Knoten)

Attribute eines Knotens vom Typ PROFIBUS Integrated

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r	modelliert	Name des Knotens.
Nodeld	String	r	modelliert	ID des Knotens.
NodeType	NetType	r	modelliert	Ein Knoten ruft seinen Typen vom Subnetz ab.
Address	String	r	dynamisch	-

Attribute eines Knotens vom Typ PROFIdrive Integrated

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Address	String	r	dynamisch	Netzwerkadresse des Knotens.
ConnectedSubnet	Subnetz	r	modelliert	Verbundenes Subnetz.
Name	String	r	modelliert	Name des Knotens.
Nodeld	String	r	modelliert	ID des Knotens.
NodeType	NetType	r	modelliert	Ein Knoten ruft seinen Typen vom Subnetz ab.

Programmcode: Attribute eines Knotens

Um die Attribute eines Knotens abzurufen oder festzulegen, ändern Sie folgenden Programmcode:

```
Node node = ...;
string nameValue = node.Name;
NetType nodeType = (NetType)node.NodeType;
node.NodeType = NetType.Mpi;
```

Programmcode: Dynamische Attribute

Um dynamische Attribute eines Knotens abzurufen oder festzulegen, ändern Sie folgenden Programmcode:

```
Node node = ...;
var attributeNames = new[]
{
    "Address", "SubnetMask", "RouterAddress", "UseRouter", "DhcpClientId",
    "IpProtocolSelection"
};
foreach (var attributeName in attributeNames)
{
    object attributeValue = ((IEngineeringObject)node).GetAttribute(attributeName);
}
```

Protokollauswahl

Wert	Beschreibung
IpProtocolSelection.None	Fehlerwert
IpProtocolSelection.Project	IP Suite innerhalb des Projekts konfiguriert.
IpProtocolSelection.Dhcp	IP Suite mit Hilfe von DHCP-Protokoll verwaltet. ID des DHCP-Client notwendig.
IpProtocolSelection.UserProgram	IP Suite mit Hilfe von FB (Funktionsbaustein) festgelegt.
IpProtocolSelection.OtherPath	IP Suite mit Hilfe von anderen Methoden festgelegt, z. B. PST Tool.
IpProtocolSelection.VialoController	IP Suite mit Hilfe von IO-Controller in Runtime festgelegt.

Netzwerktyp

Wert	Beschreibung
NetType.Asi	Netzwerktyp ist ASI.
NetType.Ethernet	Netzwerktyp ist Ethernet.
NetType.Link	Netzwerktyp ist Link.
NetType.Mpi	Netzwerktyp ist MPI.
NetType.PcInternal	Netzwerktyp ist PC internal.
NetType.Profibus	Netzwerktyp ist PROFIBUS.
NetType.ProfibusIntegrated	Netzwerktyp ist PROFIBUS Integrated.
NetType.ProfidriveIntegrated	Netzwerktyp ist PROFIdrive Integrated.
NetType.Ptp	Netzwerktyp ist PTP.
NetType.Wan	Netzwerktyp ist Wide Area Network (WAN).
NetType.Unknown	Netzwerktyp ist unbekannt.

5.8.11 Knoten mit einem Subnetz verbinden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um einen Knoten (Gerät, Schnittstelle) einem Netzwerk zuzuweisen, ändern Sie folgenden Programmcode:

```
Node node = ...;  
Subnet subnet = ...;  
node.ConnectToSubnet (subnet);
```

5.8.12 Knoten von einem Subnetz trennen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um einen Knoten (Gerät, Schnittstelle) von einem Netzwerk zu trennen, ändern Sie folgenden Programmcode:

```
Node node = ...;  
node.DisconnectFromSubnet ();
```

5.8.13 IO-System erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Ein IO-System wird durch Aufrufen der Aktion `IoController.CreateIoSystem("name")` auf einem Objekt vom Typ `IoController` erstellt. Wenn der name Null oder `String.Empty` ist, dann wird der Standardname verwendet. Der IO-Controller wird erfasst durch Zugreifen auf das Objekt des Attributs `IoControllers` an der `NetworkInterface`. Der Navigator des `IoControllers` gibt ein `IoController`-Objekt zurück.

Voraussetzungen für das Erstellen eines IO-Systems:

- Die Schnittstelle des IO-Controllers ist mit einem Subnetz verbunden.
- Der IO-Controller besitzt kein IO-System.

Programmcode

Um ein IO-System zu erstellen, ändern Sie folgenden Programmcode:

```
using System.Linq;
//...
NetworkInterface interface12 = ...;
IoSystem ioSystem = ...;
// Interface is configured as io controller
if((interface12.InterfaceOperatingMode & InterfaceOperatingModes.IoController) != 0)
{
    IoControllerComposition ioControllers = interface12.IoControllers;
    IoController ioController = ioControllers.First();
    if(ioController != null)
    {
        ioSystem = ioController.CreateIoSystem("io system");
    }
}
```

5.8.14 Attribute eines IO-Systems aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt.

Programmcode: Attribute eines IO-Systems

Um die Attribute des IoSystem abzurufen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (IIoController ioController in itf.IoControllers)
{
    IoSystem ioSystem = ioController.IoSystem;
    int ioSystemNumber = ioSystem.Number;
    string ioSystemName = ioSystem.Name;
}
```

Programmcode: Subnetz eines IO-Systems

Um zu dem Subnetz zu navigieren, das dem IO-System zugeordnet ist, ändern Sie folgenden Programmcode:

```
Subnet subnet = ioSystem.Subnet;
```

5.8.15 IO-Connector mit einem IO-System verbinden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Verwenden Sie die Aktion `ConnectToIoSystem(IoSystem ioSystem)` des `IoConnector`, um einen Profinet- oder DP-`IoConnector` mit einem vorhandenen IO-System zu verbinden.

Verwenden Sie die Aktion `GetIoController`, um zum dezentralen `IoController` zu navigieren. Weitere Informationen dazu, wie Sie zum lokalen `IoConnector` und dem IO-System navigieren, finden Sie unter `Mastersystem` oder `IO-System` einer Schnittstelle abrufen (Seite 304).

Voraussetzungen:

- Der `IoConnector` ist noch nicht mit einem IO-System verbunden.
- Die `IoConnector`-Schnittstelle ist mit dem gleichen Subnetz verbunden wie die Schnittstelle des gewünschten `IoController`.

Programmcode

Ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...;
IoSystem ioSystem = null;
foreach (IoController ioController in itf.IoControllers)
{
    ioSystem = ioController.IoSystem;
    IoConnector ioConnector = itf.IoConnectors[0];
    ioConnector.ConnectToIoSystem();
    IoController ioController2 = ioConnector.GetIoController();
}
```

5.8.16 Mastersystem oder IO-System einer Schnittstelle abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe `Verbindung zum TIA Portal aufbauen` (Seite 84)
- Ein Projekt ist geöffnet.
Siehe `Projekt öffnen` (Seite 131)

Verwendung

Der Dienst `NetworkInterface` liefert dem Navigator `IoControllers`, wovon jeder `IoController` wiederum dem Navigator `IoSystem` liefert. Das Mastersystem und das IO-System werden beide von der Klasse `IoSystem` dargestellt. Das IO-Device und der Slave werden beide `IO-Device` genannt.

- Der Navigator des `IoControllers` gibt `IoController`-Objekte zurück, wenn die Netzwerkschnittstelle ein IO-System haben kann. Im Moment wird nur ein IO-Controller zurückgegeben.
- Der Navigator des `IoConnectors` gibt `IoConnector`-Objekte zurück, wenn die Netzwerkschnittstelle als IO-Gerät mit einem IO-System verbunden werden kann. Im Moment wird nur ein IO-Connector zurückgegeben.

Programmcode: IO-System des IoController abrufen

Um das IO-System des `IoController` abzurufen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (IoController ioController in itf.IoControllers)
{
    IoSystem ioSystem = ioController.IoSystem;
    // work with the io system
}
```

Programmcode: IO-System des IoConnector abrufen

Um das IO-System des `IoConnector` abzurufen, ändern Sie folgenden Programmcode:

```
NetInterface itf = ...
foreach (IoConnector ioConnector in itf.IoConnectors)
{
    IoSystem ioSystem = ioConnector.ConnectedIoSystem;
    // work with the io system
}
```

5.8.17 IO-Controller abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Derzeit sind nur Konfigurationen mit einem IoController möglich. Ein IoController verfügt nicht über modellierte Attribute oder Aktionen.

Programmcode

Um den IO Controller abzurufen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (IoController ioController in itf.IoControllers)
{
    // work with the io controller
}
```

5.8.18 IO-Connector abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Ein IoConnector liefert die modellierten Attribute oder Aktionen.

Die folgenden Attribute und Aktionen sind am IoConnector verfügbar:

Aktionen

Aktion	Signatur	Beschreibung
ConnectToIoSystem	void ConnectToIoSystem(Siemens.Engineering.HW.IoSystem ioSystem)	Zum Verbinden eines IoConnector mit einem vorhandenen DP-Mastersystem
DisconnectFromIoSystem	void DisconnectFromIoSystem()	Zum Trennen eines IoConnector von einem vorhandenen IO-System
GetIoController	Siemens.Engineering.HW.IoController GetIoController()	Gibt den IO-Controller für diesen Connector zurück

Links

Link	Typ	Zugriff
ConnectedToIoSystem	Siemens.Engineering.HW.IoSystem	schreibgeschützt

Programmcode

Um den IO Connector abzurufen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (IoConnector ioConnector in itf.IoConnectors)
{
    // work with the IoConnector
}
```

5.8.19 IO-Connector von einem IO-System oder einem DP-Mastersystem trennen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Verwenden Sie die Aktion `DisconnectFromIoSystem()` des `IoConnector`, um einen `IoConnector` von einem vorhandenen IO-System oder einem vorhandenen DP-Mastersystem zu trennen.

Weitere Informationen dazu, wie Sie zum lokalen `IoConnector` und dem IO-System navigieren, finden Sie unter Mastersystem oder IO-System einer Schnittstelle abrufen (Seite 304).

Programmcode

Ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
IoConnector ioConnector = ...;

ioConnector.DisconnectFromIoSystem();
```

5.8.20 Attribute eines DP-Mastersystems aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Ein DP-Mastersystem bietet bestimmte Attribute, die gelesen und/oder geschrieben werden können. Die Attribute sind nur verfügbar, wenn sie im TIA Portal verfügbar sind. Der Schreibvorgang ist im Allgemeinen nur dann zulässig, wenn ein Attribut auch vom Benutzer im TIA Portal geändert werden kann. Dies kann abhängig vom DP-Master und den DP-Slaves variieren, die diesem DP-Mastersystem zugeordnet werden.

Attribute eines DP-Mastersystems

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	string	r/w	Eigenschaft	-
Number	int	r/w	Eigenschaft	Das Attribut Number übernimmt Werte, die nicht über die UI festgelegt werden können. In einem solchen Fall schlägt die Übersetzung fehl.

Programmcode: Attribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
IoSystem dpMastersystem = ...;  
string name = dpMastersystem.Name;  
int number = dpMastersystem.Number;
```

Programmcode: Attribute festlegen

Um die Attribute einzustellen, ändern Sie folgenden Programmcode:

```
IoSystem dpMastersystem = ...;  
dpMastersystem.Name = "myDpMastersystem"  
dpMastersystem.Number=42;
```


5.8.21 Auf Attribute von PN Driver zugreifen

Verwendung

Sie können mit TIA Portal Openness auf die Attribute der PROFINET-Schnittstelle von PN Driver V2.2 zugreifen.

Zugriff auf ein Attribut eines PN Driver V2.2 Submodul-Objekts

Auf die folgenden Attribute kann über die Schnittstellenoptionen mit TIA Portal Openness zugegriffen werden:

Attribute in Openness	Typ	Schreibgeschützt
DeviceReplacementWithoutExchangeableMedium	Bool	False
OverwriteDeviceNames	Bool	False
IECV22LLDPMode	Bool	False

Auf die folgenden Attribute kann über die Echtzeitoptionen mit TIA Portal Openness zugegriffen werden:

Attribute in Openness	Typ	Schreibgeschützt
CalculatedBandwidthForCyclicData	Int32	True
MaxBandwidthForCyclicData	Int32	True

Auf die folgenden Attribute kann über die Taktsynchronitätsoptionen mit TIA Portal Openness zugegriffen werden:

Attribute in Openness	Typ	Schreibgeschützt
ApplicationCycle	Int64	False
AutomaticMinimum	Bool	False
DelayTime	Double	False

Hinweis

Der Wert für DelayTime kann nur eingestellt werden, wenn der Wert für AutomaticMinimum vorher auf False gesetzt wurde.

5.8.22 Attribute eines PROFINET IO-Systems aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Ein IO-System bietet bestimmte Attribute, die gelesen und/oder geschrieben werden können. Die Attribute sind nur verfügbar, wenn sie in der UI verfügbar sind. Der Schreibvorgang ist im Allgemeinen nur dann zulässig, wenn ein Attribut auch vom Benutzer in der UI geändert werden kann. Dies kann abhängig vom IO-Controller und den IO-Devices variieren, die diesem IO-System zugeordnet werden.

Attribute eines PROFINET IO-Systems

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
MultipleUseIoSystem	Bool	r/w	dynamisch	-
Name	String	r/w	Eigenschaft	-
Number	Int	r/w	Eigenschaft	Das Attribut Number übernimmt Werte, die nicht über die UI festgelegt werden können. In einem solchen Fall schlägt die Übersetzung fehl.
UseIoSystemNameAsDeviceNameExtension	Bool	r/w	dynamisch	Wenn MultipleUseIoSystem auf TRUE gesetzt ist, dann wird UseIoSystemNameAsDeviceNameExtension auf FALSE gesetzt und der Schreibzugriff ist nicht möglich.
MaxNumberIWlanLinksPerSegment	Int	r/w	dynamisch	-

Programmcode: Attribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
string name = ioSystem.Name;
```

Programmcode: Attribute festlegen

Um die Attribute einzustellen, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
ioSystem.Name = "IOSystem_1";
```

Programmcode: Attribute bei dynamischem Zugriff abrufen

Um die Werte von dynamischen Attributen abzurufen, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
var attributeNames = new[]
{
    "MultipleUseIoSystem", "UseIoSystemNameAsDeviceNameExtension",
    "MaxNumberIWlanLinksPerSegment"
};
foreach (var attributeName in attributeNames)
{
    object attributeValue = ((IEngineeringObject)ioSystem).GetAttribute(attributeName);
}
```

Programmcode: Attribute bei dynamischem Zugriff festlegen

Um die Werte der dynamischen Attribute festzulegen, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
((IEngineeringObject)ioSystem).SetAttribute("MultipleUseIoSystem", true);
```

5.8.23 DP-Mastersystem löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode: PROFINET IO-System löschen

Um ein PROFINET IO-System zu löschen, ändern Sie den folgenden Programmcode:

```
IoController ioController = ...;
IoSystem ioSystem = ioController.IoSystem;

ioSystem.Delete();
```

5.8.24 Profinet IO-System löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um ein Profinet IO-System zu löschen, ändern Sie den folgenden Programmcode:

```
IoController ioController = ...;  
IoSystem ioSystem = ioController.IoSystem;  
ioSystem.Delete();
```

5.8.25 DP-Mastersystem erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Ein DP-Mastersystem wird durch Aufrufen der Aktion `CreateIoSystem(string nameOfIoSystem)` auf einem Objekt vom Typ `IoController` erstellt. Der IO-Controller wird erfasst durch Zugreifen auf das Objekt des Attributs `IoControllers` an der `NetworkInterface`.

Voraussetzungen für das Erstellen eines DP-Mastersystems:

- Die Schnittstelle des IO-Controllers ist mit einem Subnetz verbunden.
- Der IO-Controller besitzt kein IO-System.

Programmcode

Um ein DP-Mastersystem zu erstellen, ändern Sie folgenden Programmcode:

```
//...
NetworkInterface interfacel2 = ...;
IoSystem dpMasterSystem = ...;
// Interface is configured as master or as master and slave
if((interfacel2.InterfaceOperatingMode & InterfaceOperatingModes.IoController) != 0)
{
    IoControllerComposition ioControllers = interfacel2.IoControllers;
    IoController ioController = ioControllers.First();
    if(ioController != null)
    {
        dpMasterSystem = ioController.CreateIoSystem("dp master system");
    }
}
```

5.8.26 Port-Verschaltungsinformationen des Port-Geräteelements aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

NetworkPort bietet die Verknüpfung ConnectedPorts, die eine Enumeration von Ports ist, um auf alle verschalteten Partnerports eines Ports zuzugreifen.

Es können nur Ports miteinander verschaltet werden, die auch in der TIA UI verschaltet werden können. Beispielsweise können zwei Ports derselben Ethernet-Schnittstelle nicht miteinander verschaltet werden. Eine wiederherstellbare Ausnahme wird ausgelöst,

- wenn bereits eine Verschaltung mit demselben Partnerport besteht
- wenn Sie versuchen, zwei Ports zu verschalten, die nicht miteinander verschaltet werden können
- wenn Sie versuchen, eine zweite Verschaltung zu einem Port anzulegen, der keine alternativen Partner unterstützt

Programmcode: Portverschaltung abrufen

Um die Port-Verschaltungsinformationen eines Port-Geräteelements abzurufen, ändern Sie den folgenden Programmcode:

```
NetworkPort port = ...;
foreach (NetworkPort partnerPort in port.ConnectedPorts)
{
    // work with the partner port
}
```

Programmcode: Portverschaltungen erstellen

Ändern Sie folgenden Programmcode:

```
NetworkPort port1 = ...;
NetworkPort port2 = ...;
port1.ConnectToPort(port2);

// port supports alternative partners
NetworkPort port1 = ...;
NetworkPort port2 = ...;
NetworkPort port3 = ...;
port1.ConnectToPort(port2);
port1.ConnectToPort(port3);
```

Programmcode: Portverschaltung löschen

Ändern Sie folgenden Programmcode:

```
NetworkPort port1 = ...;
NetworkPort port2 = ...;
port1.DisconnectFromPort(port2);
```

5.8.27 Attribute der Port-Verschaltung

Attribute für Port-Verschaltungen

TIA Portal Openness unterstützt die folgenden Attribute für Port-Verschaltungen. Wenn die Attribute in der Benutzeroberfläche verfügbar sind, sind die entsprechenden Attribute auch über TIA Portal Openness zugänglich. Wenn der Anwender Zugriff hat und die Attribute in der Benutzeroberfläche ändern kann, können die Attribute auch über TIA Portal Openness geändert werden.

Attributname	Datentyp	Schreibbar	Zugriff	Beschreibung
MediumAttachment-Type	MediumAttachment-Type	Schreibgeschützt	Dynamisches Attribut	-
CableName	CabelName	Lesen-Schreiben	Dynamisches Attribut	-
AlternativePartnerPorts	Bool	Lesen-Schreiben	Dynamisches Attribut	Nur verfügbar, wenn die Funktionalität des Werkzeugwechslers unterstützt wird.
SignalDelaySelection	SignalDelaySelection	Lesen-Schreiben	Dynamisches Attribut	-
CableLength	CableLength	Lesen-Schreiben	Dynamisches Attribut	-
SignalDelayTime	Double	Lesen-Schreiben	Dynamisches Attribut	-

Enum-Werte von Port-Verschaltungsattributen

Die Enum MediumAttachmentType hat folgende Werte.

Wert	Beschreibung
MediumAttachmentType.None	Anbindungstyp kann nicht ermittelt werden.
MediumAttachmentType.Copper	Anbindungstyp ist Kupfer.
MediumAttachmentType.FiberOptic	Anbindungstyp ist Glasfaser.

Die Enum CableName hat folgenden Wert

Wert	Beschreibung
CableName.None	Kein Kabelname angegeben
CableName.FO_Standard_Cable_9	FO-Standardkabel GP (9 µm)
CableName.Flexible_FO_Cable_9	Flexibles FO-Kabel (9 µm)
CableName.FO_Standard_Cable_GP_50	FO-Standardkabel GP (50 µm)
CableName.FO_Trailing_Cable_GP	FO-Schleppkabel / GP
CableName.FO_Ground_Cable	FO-Erdungskabel
CableName.FO_Standard_Cable_62_5	FO-Standardkabel (62,5 µm)
CableName.Flexible_FO_Cable_62_5	Flexibles FO-Kabel (62,5 µm)
CableName.POF_Standard_Cable_GP	POF-Standardkabel GP
CableName.POF_Trailing_Cable	POF-Schleppkabel
CableName.PCF_Standard_Cable_GP	PCF-Schleppkabel / GP
CableName.GI_POF_Standard_Cable	GI-POF-Standardkabel
CableName.GI_POF_Trailing_Cable	GI-POF-Schleppkabel
CableName.GI_PCF_Standard_Cable	GI-PCF-Standardkabel
CableName.GI_PCF_Trailing_Cable	GI-PCF-Schleppkabel
CableName.GI_POF_Standard_Cable	GI-POF-Standardkabel

Wert	Beschreibung
CableName.GI_POF_Trailing_Cable	GI-POF-Schleppkabel
CableName.GI_PCF_Standard_Cable	GI-PCF-Standardkabel
CableName.GI_PCF_Trailing_Cable	GI-PCF-Schleppkabel

Die Enum SignalDelaySelection hat folgende Werte.

Wert	Beschreibung
SignalDelaySelection.None	-
SignalDelaySelection.CableLength	CableLength dient zur Festlegung der Signalverzögerung.
SignalDelaySelection.SignalDelayTime	CableDelayTime dient zur Festlegung der Signalverzögerung.

Die Enum CableLength hat folgende Werte

Wert	Beschreibung
CableLength.None	Kabellänge ist nicht angegeben.
CableLength.Length20m	Kabellänge ist 20 m.
CableLength.Length50m	Kabellänge ist 50 m.
CableLength.Length100m	Kabellänge ist 100 m.
CableLength.Length1000m	Kabellänge ist 1000 m.
CableLength.Length3000m	Kabellänge ist 3000 m.

Attribute von Port-Optionen

Die Attribute von Port-Optionen werden nachfolgend dargestellt.

Attributname	Datentyp	Schreibbar	Zugriff
PortActivation	Bool	Lesen-Schreiben	Dynamisches Attribut
TransmissionRateAnd-Duplex	TransmissionRateAnd-Duplex	Lesen-Schreiben	Dynamisches Attribut
PortMonitoring	Bool	Lesen-Schreiben	Dynamisches Attribut
TransmissionRateAuto-Negotiation	Bool	Lesen-Schreiben	Dynamisches Attribut
EndOfDetectionOfAccessibleDevices	Bool	Lesen-Schreiben	Dynamisches Attribut
EndOfTopologyDiscovery	Bool	Lesen-Schreiben	Dynamisches Attribut
EndOfSyncDomain	Bool	Lesen-Schreiben	Dynamisches Attribut

Die Enum TransmissionRateAndDuplex hat folgende Werte.

Wert	Beschreibung
TransmissionRateAndDuplex.None	-
TransmissionRateAndDuplexAutomatic	Automatisch
TransmissionRateAndDuplex.AUI10Mbps	10 Mbps AUI
TransmissionRateAndDuplex.TP10MbpsHalfDuplex	TP 10 Mbps Halbduplex
TransmissionRateAndDuplex.TP10MbpsFullDuplex	TP 10 Mbps Vollduplex
TransmissionRateAndDuplex.AsyncFiber10MbpsHalfDuplex	Asynchrone Glasfaser 10 Mbit/s Halbduplexmodus
TransmissionRateAndDuplex.AsyncFiber10MbpsFullDuplex	Asynchrone Glasfaser 10 Mbit/s Vollduplexmodus
TransmissionRateAndDuplex.TP100MbpsHalfDuplex	TP 100 Mbps Halbduplex
TransmissionRateAndDuplex.TP100MbpsFullDuplex	TP 100 Mbps Vollduplex
TransmissionRateAndDuplex.FO100MbpsFullDuplex	FO 100 Mbps Vollduplex
TransmissionRateAndDuplex.X1000MbpsFullDuplex	X1000 Mbps Vollduplex
TransmissionRateAndDuplex.FO1000MbpsFullDuplexLD	FO 1000 Mbps Vollduplex LD
TransmissionRateAndDuplex.FO1000MbpsFullDuplex	FO 1000 Mbps Vollduplex
TransmissionRateAndDuplex.TP1000MbpsFullDuplex	TP 1000 Mbps Vollduplex
TransmissionRateAndDuplex.FO10000MbpsFullDuplex	FO 10000 Mbps Vollduplex
TransmissionRateAndDuplex.FO100MbpsFullDuplexLD	FO 100 Mbps Vollduplex LD
TransmissionRateAndDuplex.POFPCF100MbpsFullDuplexLD	POF/PCF 100 Mbps Vollduplex

5.8.28 Attribute eines Ports aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Ein Geräteelement, das zugleich ein Port ist, bietet gegenüber einem einfachen Geräteelement zusätzliche Funktionalität.

- Es ist möglich, auf die verknüpften Partnerports des Ports zuzugreifen.
- Es ist möglich, auf die Schnittstelle des Ports zuzugreifen.

Um auf diese zusätzliche Funktionalität zuzugreifen, muss die NetworkPort-Funktion, ein bestimmter Dienst des Geräteelements, verwendet werden.

Programmcode: Auf einen Port zugreifen

Um auf Attribute eines Kanals zuzugreifen, ändern Sie folgenden Programmcode:

```
NetworkPort port = ((IEngineeringServiceProvider) deviceItem).GetService<NetworkPort>();
if (port != null)
{
    ... // Work with the port
}
```

Attribute eines Ports

Ein Port hat folgende Attribute:

```
NetworkPort port = ...;
var connectedPorts = port.ConnectedPorts;
var myInterface = port.Interface;
```

5.8.29 DP-Mastersysteme eines Subnetzes enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Die Enumeration von IoSystem stellt alle DP-Mastersysteme, die sich auf dem Subnetz befinden, bereit. Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt.

Programmcode

Um DP-Mastersysteme aus dem Subnetz zu enumerieren, ändern Sie folgenden Programmcode:

```
Subnet subnet = ...;
foreach (IoSystem ioSystem in subnet.IoSystems)
{
    // work with the io system
}
```

5.8.30 Zugeordnete IO-Connectors enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt.

Wird verwendet für:

- Enumeration zugeordneter IO-Connectors des DP-Mastersystems
- Enumeration zugeordneter IO-Connectors des Profinet-IO-Systems

Programmcode

Um zugeordnete IO Connectors des DP-Mastersystems zu enumerieren, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
foreach (IoConnector ioConnector in ioSystem.ConnectedIoDevices)
{
    // work with the io connector
}
```

5.8.31 DP-IO-Connector mit einem DP-Mastersystem verbinden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Verwenden Sie die Aktion `ConnectToIoSystem(ioSystem)` des `IoConnector`, um einen `IoConnector` mit einem vorhandenen DP-Mastersystem zu verbinden.

Verwenden Sie die Aktion `GetIoController`, um zum dezentralen `IoController` zu navigieren. Weitere Informationen dazu, wie Sie zum lokalen `IoConnector` und dem IO-System navigieren, finden Sie unter Mastersystem oder IO-System einer Schnittstelle abrufen (Seite 304).

Voraussetzungen:

- Der `IoConnector` ist noch nicht mit einem IO-System verbunden.
- Die `IoConnector`-Schnittstelle ist mit dem gleichen Subnetz verbunden wie die Schnittstelle des gewünschten `IoController`.

Programmcode

Ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
IoConnector ioConnector = ...;
ioConnector.ConnectToIoSystem(ioSystem);
IoController ioController = ioConnector.GetIoController();
```

5.8.32 Zugriff auf AS-i-Profil und Parameterattribute für virtuelle Slaves

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Das TIA Portal Openness unterstützt die folgenden zusätzlichen Parameter des AS-i-Profiles für die virtuellen Slaves des CTT5 AS-i Slave unter Verwendung der StructuredData-Namen:

Name	Beschreibung
AsiProfileVirtualSlave1	Enthält die AS-i Profilparameter für den virtuellen Slave 1
AsiProfileVirtualSlave2	Enthält die AS-i Profilparameter für den virtuellen Slave 2
AsiProfileVirtualSlave3	Enthält die AS-i Profilparameter für den virtuellen Slave 3

Die AS-i-Slave-Parameter für die virtuellen Slaves sind im Folgenden aufgeführt:

Name	Beschreibung
AsiParameterVirtualSlave1	Enthält die AS-i-Parameter für den virtuellen Slave 1
AsiParameterVirtualSlave2	Enthält die AS-i-Parameter für den virtuellen Slave 2
AsiParameterVirtualSlave3	Enthält die AS-i-Parameter für den virtuellen Slave 3

Programmcode

Ändern Sie folgenden Programmcode, um die zusätzlichen Attribute von AS-i-Slaves abzurufen und einzustellen:

```
DeviceItem slaveModule = ...;
var structuredDataNamesProfile = new[]{"AsiProfileVirtualSlave1",
"AsiProfileVirtualSlave2",
"AsiProfileVirtualSlave3"};
var structuredDataNamesAsiParameter = new[]{"AsiParameterVirtualSlave1",
"AsiParameterVirtualSlave2",
"AsiParameterVirtualSlave3"};
var attributeNamesProfile = new[]{ "AsiProfileID", "AsiProfileIO", "AsiProfileID2",
"AsiProfileID1" };

string attributeNameAsiParameter = "AsiSlaveParameter";
foreach (var structuredDataName in structuredDataNamesProfile)
{
    foreach (var attributeName in attributeNamesProfile)
    {
        StructuredData structuredData =
        (StructuredData)slaveModule.GetAttribute(structuredDataName);

        //get
        UInt32 attributeValue = (UInt32)structuredData.GetAttribute(attributeName);
    }
    //set
    slaveModule.SetAttribute(attributeName, (UInt32)5);
}
foreach (var structuredDataName in structuredDataNamesAsiParameter)
{
    StructuredData structuredData =
    (StructuredData)slaveModule.GetAttribute(structuredDataName);
    //get
    UInt32 attributeValue = (UInt32)structuredData.GetAttribute(attributeNameAsiParameter);
    //set
    slaveModule.SetAttribute(attributeName, (UInt32)5);
}
```

Siehe auch

[Verbindung zum TIA Portal aufbauen \(Seite 84\)](#)

[Projekt öffnen \(Seite 131\)](#)

5.8.33 Zugriff auf CM DP als DP-Slave und Transferbereich

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Wenn ein CM DP gesteckt ist, können Sie mit TIA Portal Openness eine ET200SP PLC als DP-Slave konfigurieren. Außerdem ist es möglich, in TIA Portal Openness Transferbereiche an der DP-Schnittstelle zu erstellen, zu konfigurieren und zu löschen. Die Handhabung ist ähnlich der Handhabung von Transferbereichen an der PN-Schnittstelle, die unter Openness-Transferbereiche für PN/PN-Koppler (Seite 330) beschrieben wird.

Die folgenden dynamischen Attribute werden an der DP-Schnittstelle des Geräteelements unterstützt:

Attributname	Datentyp	Zugriff	Schreibbar
DpUseForTestCommissioningRouting	Boolean	nur verfügbar, wenn als DP-Slave konfiguriert	r/w
DpWatchdog	Boolean	nur verfügbar, wenn als DP-Slave konfiguriert und DP-Master zugewiesen	r/w

Programmcode: Transferbereiche erstellen

Ändern Sie folgenden Programmcode, um an der DP-Schnittstelle einen Transferbereich zu erstellen:

```
NetworkInterface cpuItf = CpuInterface.GetService<NetworkInterface>();
// Create TransferAreas
TransferAreaComposition transferAreas = cpuItf.TransferAreas;
// Simple TransferArea
TransferArea transferAreaExample = transferAreas.Create("Example TA MS",
TransferAreaType.MS);
```

Parameter von Transferbereichen

Die folgenden Parameter werden in Transferbereichen der DP-Slave-Konfiguration unterstützt:

Parametername	Datentyp	Zugriff
Name	string	read/write
Direction	enum	read/write
Comment	string	read/write
LocalToPartnerLength	Int32	read/write

Parametername	Datentyp	Zugriff
PartnerToLocalLength	Int32	read/write
LocalAddresses	AddressComposition object	read
PartnerAddresses	AddressComposition object	read
PositionNumber	Int32	read
Type	enum	read

Programmcode: Transferbereich löschen

Ändern Sie folgenden Programmcode so, dass ein Transferbereich für die DP-Slave-Konfiguration gelöscht wird:

```
TransferArea transferAreaExample = transferAreas.Create("Example TA MS",
TransferAreaType.MS);
transferAreaExample.Delete();
```

Siehe auch

Projekt öffnen (Seite 131)

5.8.34 Kopplung 1516 taktsynchron zentral und dezentral Profinet

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness folgende Vorgänge ausführen:

- Bei einer PLC 1516 mit einem taktsynchronfähigen Modul und der Möglichkeit, die PLC im taktsynchronen Betrieb einzustellen
- Bei der vorherigen Konfiguration, bei der das Modul OB61 zugewiesen wird und der Möglichkeit, das Modul in den taktsynchronen Betrieb zu versetzen
- Bei der vorherigen Konfiguration mit zusätzlicher, der PLC zugewiesener ET200SP und einem im taktsynchronen Betrieb konfigurierten Modul, das demselben OB61 zugewiesen ist, wobei die Sendclock-Quelle eingestellt werden kann (PLC und dezentral werden gekoppelt, Sendclock-Quelle wird auf PN X1 gesetzt).

Attribute

Attributname	Datentyp	Schreibbar	Verfügbarer Bereich / Verfügbare Werte	Voraussetzungen	Beschreibung	Zugriff
IsochronousMode	bool	r/w	Wahr/Falsch	Taktsynchronfähiges Gerät	Abrufen/Einstellen des taktsynchronen Betriebs	Dynamisches Attribut
SourceCycleTime	int	r/w	Gerätespezifisch	Taktsynchroner Betrieb ist aktiviert	Abrufen/Einstellen der Sendclock-Quelle	Dynamisches Attribut
SendClock	float	r/w	Gerätespezifisch	Taktsynchroner Betrieb ist aktiviert	Abrufen/Einstellen von Sendclock	Dynamisches Attribut
IsochronousTiToCalculationMode	enum	r/w	FromOB, AutomaticMinimum, Manual	Taktsynchroner Betrieb ist aktiviert	Abrufen/Einstellen der Ti/To-Werte	Dynamisches Attribut
IsochronousTi	double	r/w	Gerätespezifisch	Taktsynchroner Betrieb ist aktiviert	Abrufen/Einstellen der Zeit Ti	Dynamisches Attribut
IsochronousTo	double	r/w	Gerätespezifisch	Taktsynchroner Betrieb ist aktiviert	Abrufen/Einstellen der Zeit To	Dynamisches Attribut

Programmcode: Beispiele mit S71500/ET200MP-Station

Ändern Sie den folgenden Programmcode für die PLC, die eine S71500/ET200MP-Station repräsentiert:

```
//plc represents an S71500/ET200MP station
//Getting headmodule of the station
DeviceItem plcHeadModule = plc.DeviceItems[1];
//Getting and I/O module of the station
DeviceItem module = plc.DeviceItems[2];
```

Ändern Sie den folgenden Programmcode, um die Eigenschaften des taktsynchronen Betriebs auf dem PLC-Kopfmodul einzustellen:

```
//Turn on isochronous mode
plcHeadModule.SetAttribute("IsochronousMode", true);
//Set source of sendclock to "Use send clock of PROFINET interface [X1]"
plcHeadModule.SetAttribute("SourceCycleTime", 1);
//Set source of sendclock to "Local send clock"
plcHeadModule.SetAttribute("SourceCycleTime", 0);
//Set sendclock
plcHeadModule.SetAttribute("SendClock", (float)2.1);
//Set calculation mode to Automatic minimum
plcHeadModule.SetAttribute("IsochronousTiToCalculationMode",
IsochronousTiToCalculationMode.AutomaticMinimum);
//Set calculation mode to Manual
plcHeadModule.SetAttribute("IsochronousTiToCalculationMode",
IsochronousTiToCalculationMode.Manual);
//Set Time Ti
plcHeadModule.SetAttribute("IsochronousTi", (double)0.55);
//Set Time To
plcHeadModule.SetAttribute("IsochronousTo", (double)0.65);
```

Ändern Sie den folgenden Programmcode, um die Eigenschaften des taktsynchronen Betriebs auf I/O-Modulen einzustellen:

```
//Isochronous relevant properties can be found on the Address of the submodule
DeviceItem subModule = module.DeviceItems[0];
Address address = subModule.Addresses[0];
//Turn on isochronous mode
address.SetAttribute("IsochronousMode", true);
//Set Process image to 3
address.SetAttribute("ProcessImage", 3);
```

Programmcode: SIMATIC Drive Controller

Das Einstellen der Eigenschaften des taktsynchronen Betriebs auf dem PLC-Kopfmodul ist bei einer S71500/ET200MP-Station identisch.

Ändern Sie den folgenden Programmcode, um die Eigenschaften des taktsynchronen Betriebs der Zentralbaugruppe (X142) einzustellen:

Ändern Sie den folgenden Programmcode, um die Eigenschaften des taktsynchronen Betriebs eines integrierten Telegramms einzustellen:

5.8.35 Openness für CP 1604/CP 1616/CP 1626

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Vor dem Übersetzen des Projekts müssen alle Geräte "offline" sein.

Einleitung

Mi Hilfe der TIA Portal Openness-Anwendung können Sie die Transferbereiche und die Zuordnungsregeln der Transferbereiche konfigurieren. Dies gilt für die Kommunikationsprozessoren CP 1604/CP 1616 ab V2.8 (und je nach Artikelnummer auch ab V2.7) sowie CP 1626 ab V1.1.

Konfiguration von Transferbereichen

Transferbereiche erstellen

Um einen Transferbereich vom Typ "CD" für einen CP 1604 zu erstellen, ändern Sie folgenden Programmcode:

```
NetworkInterface cpItf = CP 1604Interface.GetService<NetworkInterface>();
//Create Transfer Areas
TransferAreaComposition transferAreas = cpItf.TransferAreas;
//Simple TransferArea of type Input
TransferArea transferAreaInput = transferAreas.Create("Input CD", TransferAreaType.CD);
```

Die folgenden Attribute werden zur Erstellung von Transferbereichen verwendet:

Attribut	Beschreibung
name	Gibt den Namen des zu erstellenden Transferbereichs an.
type	Gibt den Typ des zu erstellenden Transferbereichs an. Folgende Typen sind möglich:
	TransferAreaType.CD Steuerungsgerät für Datenaustausch
	TransferAreaType.F_PS Datenaustausch PROFIsafe
	TransferAreaType.TM Zuordnung des Transfermoduls
	Hinweis: Es ist nicht möglich, den Typ später zu ändern.

Attribute der Transferbereiche festlegen

Um die Attribute eines Transferbereichs festzulegen, ändern Sie folgenden Programmcode:

```
transferAreaTm.LocalToPartnerLength = 8;
transferAreaTm.Direction = TransferAreaDirection.LocalToPartner;
string name = transferAreaTm.Name
```

Einige Attribute müssen festgelegt oder abgefragt werden, alle Attribute können jedoch mit den allgemeinen Aufrufen "GetAttribute()" oder "SetAttribute()" festgelegt oder abgefragt werden. Ändern Sie dazu folgenden Programmcode:

```
const string myIndividualComment = "MyIndividualComment";
transferAreaTm.SetAttribute("Comment", myIndividualComment);
Int32 updateTime = transferAreaTm.GetAttribute("TransferUpdateTime");
```

Die folgenden Attribute können für Transferbereiche festgelegt werden:

Attribut	Beschreibung
Name (string)	Gibt den Namen des Transferbereichs an.
Direction	Gibt die Richtung an, in der die Daten des Transferbereichs übertragen werden. Folgende Richtungen sind möglich:
	TransferAreaDirection.LocalToPartner Daten des Transferbereichs werden vom IO-Device an die überlagerte Steuerung übertragen.
	TransferAreaDirection.partnerToLocal Daten des Transferbereichs werden von der überlagerten Steuerung an das IO-Device übertragen.
	TransferAreaDirection.bidirectional Daten des Transferbereichs können zwischen der überlagerten Steuerung und dem IO-Device in beiden Richtungen übertragen werden. Mit dem Attribut "LocalToPartnerLength" wird die Länge angegeben, mit der Daten des Transferbereichs vom IO-Device an die überlagerte Steuerung übertragen werden. Das Attribut "PartnerToLocalLength" legt die Datenmenge fest, die von der überlagerten Steuerung an das IO-Device übertragen werden kann.
Comment (string)	Textfeld für Kommentar zum Transferbereich.
LocalToPartnerLength	Gibt die Länge der Daten des Transferbereichs an, die vom IO-Device an die überlagerte Steuerung übertragen werden.
PartnerToLocalLength	Gibt die Länge der Daten des Transferbereichs an, die von der überlagerten Steuerung an das IO-Device übertragen werden.
LocalAdresses	Gibt die Eingangs- und Ausgangsadressen des Transferbereichs im lokalen Gerät an.
PartnerAdresses	Gibt die Eingangs- und Ausgangsadressen des Transferbereichs in der überlagerten Steuerung an.
TransferUpdateTime(Int32)	Gibt die Aktualisierungszeit des Transferbereichs an. Nur bei einem Transferbereich vom Typ "TransferAreaType.TM" festgelegt oder abgefragt.
PositionNumber	Gibt die Anzahl der virtuellen Submodule dieses Transferbereichs an.
Type	Gibt den Typ des Transferbereichs an, schreibgeschützt.
TransferAreaMappingRules	Gibt die Routing-Tabelle des Routing-Bereichs an, schreibgeschützt.

Transferbereiche löschen

Ändern Sie zum Löschen von Transferbereichen folgenden Programmcode:

```
transferAreaInput.Delete();
```

Iteration über Transferbereiche

Ändern Sie zum Iterieren von Transferbereichen folgenden Programmcode:

```
TransferAreaComposition transferAreas = cpItf.TransferAreas;
foreach (TransferArea transferArea in transferAreas)
{
    transferArea.Delete();
}
```

Die folgenden Attribute können für das E/A-Routing festgelegt werden:

Attribut	Beschreibung
Offset	In Bit angegebener Offset im Routing-Bereich, dem die Daten zugewiesen werden sollen. Die Länge des Offsets laut Festlegung durch die Attribute "Begin" und "End".
Target	Gibt das Modul oder Submodul des IO-Device an, das die Daten enthält, die der Konfiguration des IO-Device zugeordnet werden sollen, d. h. einen Transferbereich vom Typ "TM".
IoType	Das Attribut "IoType" kann nur in einem Transferbereich vom Typ "Input" geändert werden. Zusätzlich muss ein gemischtes Modul für diesen Transferbereich als "Ziel" konfiguriert werden. Erst kann können Sie auswählen, ob die Daten der Eingänge (IoType.Input) zu lesen sind oder ob die Daten der Ausgänge (IoType.Output) (zurück-)gelesen werden sollen.
Begin	Gibt den Anfang der mit dem Attribut "Target" zu lesenden Daten an.
End	Gibt das Ende der mit dem Attribut "Target" zu lesenden Daten an.

Konfiguration des E/A-Routing

E/A-Routen erstellen

Ändern Sie zum Erstellen von E/A-Routen folgenden Programmcode:

```
//Create TransferAreaMappingRule
TransferAreaMappingRuleComposition routingTable = transferArea.TransferAreaMappingRules;///
//Create a new IO route
TransferAreaMappingRule route1 = routingTable.Create();
```

Attribute von E/A-Routen festlegen

Die folgenden Attribute können für das E/A-Routing festgelegt werden:

Attribut	Beschreibung
Offset	In Bit angegebener Offset im Routing-Bereich, dem die Daten zugewiesen werden sollen. Die Länge des Offsets laut Festlegung durch die Attribute "Begin" und "End".
Target	Gibt das Modul oder Submodul des IO-Device an, das die Daten enthält, die der Konfiguration des IO-Device zugeordnet werden sollen, d. h. einen Transferbereich vom Typ "TM".

Attribut	Beschreibung
IoType	Das Attribut "IoType" kann nur in einem Transferbereich vom Typ "Input" geändert werden. Zusätzlich muss ein gemischtes Modul für diesen Transferbereich als "Target" konfiguriert werden. Erst dann können Sie auswählen, ob die Daten der Eingänge (IoType.Input) zu lesen sind oder ob die Daten der Ausgänge (IoType.Output) (zurück-)gelesen werden sollen.
Begin	Gibt den Anfang der mit dem Attribut "Target" zu lesenden Daten an.
End	Gibt das Ende der mit dem Attribut "Target" zu lesenden Daten an.

E/A-Routen löschen

Ändern Sie zum Löschen von E/A-Routen folgenden Programmcode:

```
transferAreaMappingRule.Delete();
```

Iteration über E/A-Routing

Ändern Sie für eine Iteration über E/A-Routing folgenden Programmcode:

```
TransferAreaMappingRuleComposition routingTable = transferArea.TransferAreaMappingRules;
foreach (TransferAreaMappingRule route in routingTable)
{
    route.Delete();
}
```

5.8.36 Openness-Transferbereiche für PN/PN-Koppler

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Vor dem Übersetzen des Projekts müssen alle Geräte "offline" sein.

Einleitung

Mit der TIA Portal Openness-Anwendung können Sie Module und Submodule hinzufügen sowie Transferbereiche für PN/PN-Koppler löschen und suchen.

Konfiguration von Transferbereichen

Transferbereiche erstellen

Um beispielsweise ein Modul an der nächsten freien Position zu erstellen, verwenden Sie folgenden Programmcode:

```
NetworkInterface coupler_Itf = deviceItem.GetService<NetworkInterface>();  
TransferAreaComposition transferAreas = coupler_Itf.TransferAreas;  
// Create a transfer area of type input at next free positionnumber  
TransferArea transferAreaInput = transferAreas.Create("Input", TransferAreaType.IN);
```

Hinweis

Es ist nicht möglich, ein Submodul ohne Positionsnummer zu erstellen.

Um beispielsweise ein Modul an einer Positionsnummer und ein Submodul an der nächsten freien Submodul-Positionsnummer zu erstellen, verwenden Sie folgenden Programmcode:

```
// Create a transfer area of type input at positionnumber 5  
TransferArea transferAreaInput = transferAreas.Create("Input", TransferAreaType.IN, 5);
```

Die folgenden Attribute werden zur Erstellung von Transferbereichen verwendet:

Attribut	Beschreibung
Type	Gibt den Typ des zu erstellenden Transferbereichs an. Folgende Typen sind möglich:
TransferAreaType.None	Standardwert
TransferAreaType.IN	Eingang des Transferbereichs
TransferAreaType.OUT	Ausgang des Transferbereichs
TransferAreaType.MSI	Gemeinsamer Eingang des Transferbereichs (MSI)
TransferAreaType.MSO	Gemeinsamer Ausgang des Transferbereichs (MSO)
TransferAreaType.MSO_LOCAL	Gemeinsamer lokaler Ausgang des Transferbereichs (MSO_LOCAL)
TransferAreaType.RECORD_WRITE_STO	Transferbereich für Record Data Write, bis zu acht Datensätze werden gepuffert
TransferAreaType.RECORD_WRITE_PUB	Transferbereich für Record Data Write überschreibt vorherigen Datensatz
TransferAreaType.RECORD_READ_STO	Transferbereich für Record Data Write liest den ältesten Datensatz im Puffer
TransferAreaType.RECORD_READ_PUB	Transferbereich für Record Data Write liest den zuletzt geschriebenen Datensatz
TransferAreaType.MSI_MSO	Gemeinsamer Eingang des Transferbereichs (MSI) / Gemeinsamer Ausgang des Transferbereichs (MSO)
TransferAreaType.IN_OUT	Eingang des Transferbereichs / Ausgang des Transferbereichs
TransferAreaType.LOCAL_RECORD_STO	Transferbereich für lokale Datensatzkopplung, bis zu acht Datensätze werden gepuffert
TransferAreaType.LOCAL_RECORD_PUB	Transferbereich für lokale Datensatzkopplung, vorheriger Datensatz wird überschrieben
TransferAreaType.SUB_MSI	Transferbereich für gemeinsamen Eingang (MSI) kopieren
TransferAreaType.SUB_MSO	Transferbereich für gemeinsamen Ausgang (MSO) kopieren
TransferAreaType.SUB_LOCAL_RECORD_STO_READ	Transferbereich für Record Data Read zum Lesen des ältesten Datensatzes im Puffer
TransferAreaType.SUB_LOCAL_RECORD_PUB_READ	Transferbereich für Record Data Read zum Lesen des zuletzt geschriebenen Datensatzes
TransferAreaType.PROFISA_FE_IN12_OUT6	Transferbereich mit 12-Byte-Eingang und 6-Byte-Ausgang
TransferAreaType.PROFISA_FE_IN6_OUT12	Transferbereich mit 6-Byte-Eingang und 12-Byte-Ausgang

Attribute der Transferbereiche festlegen

Um die Attribute eines Transferbereichs festzulegen, ändern Sie z. B. folgenden Programmcode:

```
// Change input length
CouplerTransf.LocalToPartnerLength = 5;
// Change output length
CouplerTransf.PartnerToLocalLength = 5;
// Change name
CouplerTransf.Name = "Testname";
```

Einige Attribute müssen festgelegt oder abgefragt werden, alle Attribute können jedoch mit den allgemeinen Aufrufen "TransferArea.GetAttribute()" oder "SetAttribute()" festgelegt oder abgefragt werden. Verwenden Sie z. B. folgenden Programmcode:

```
// Set Comment of Transferarea
string myIndividualComment = "MyIndividualComment";
CouplerTransf.SetAttribute("Comment", myIndividualComment);
// Get positionNumber of Transferarea
Int32 positionNumber = (Int32)CouplerTransf.GetAttribute("PositionNumber");
```

Die folgenden Attribute können für Transferbereiche festgelegt werden:

Attribut	Beschreibung	
Name (string)	Gibt den Namen des Transferbereichs an.	
Direction	Gibt die Richtung an, in der die Daten des Transferbereichs übertragen werden. Folgende Richtungen sind möglich:	
	TransferAreaDirection.LocalTo- Partner	Daten des Transferbereichs werden vom IO-Device an die überlagerte Steuerung übertragen.
	TransferAreaDirection.partnerTo- Local	Daten des Transferbereichs werden von der überlagerten Steuerung an das IO-Device übertragen.
	TransferAreaDirection.bidi- rectional	Daten des Transferbereichs können zwischen der überlagerten Steuerung und dem IO-Device in beiden Richtungen übertragen werden. Mit dem Attribut "LocalToPartnerLength" wird die Länge angegeben, mit der Daten des Transferbereichs vom IO-Device an die überlagerte Steuerung übertragen werden. Das Attribut "PartnerToLocal- Length" legt die Datenmenge fest, die von der überlagerten Steuerung an das IO-Device übertragen werden kann.
Comment (string)	Textfeld für Kommentar zum Transferbereich	
LocalToPartnerLength	Gibt die Eingangsdatenlänge des Transferbereichs an	
PartnerToLocalLength	Gibt die Ausgangsdatenlänge des Transferbereichs an.	
PositionNumber	Gibt die Steckplatznummer des Transferbereichs an. Es ist nicht möglich, ein Submodul ohne Positionsnummer zu erstellen	
ExtendedPositionNumber	Gibt die Untersteckplatznummer des Transferbereichs an Die erweiterte Positionsnummer wird nur für Transferbereiche mit mehreren Submodulen wie MSI benötigt	
Type	Gibt den Typ des Transferbereichs an	
SharedDeviceAccessConfigured	Steuert Zugriff auf PLC	

Attribut	Beschreibung
UpdateAlarm	Gibt den Aktualisierungsalarm des Transferbereichs an
RecordIndex	Legt die Datensatznummer fest, die Sie beim Schreiben des Datensatzes verwenden

Transferbereiche löschen

Verwenden Sie zum Löschen von Transferbereichen folgenden Programmcode:

```
TransferArea TransferAreaInput = transferAreas.Create("Input", TransferAreaType.IN);  
TransferAreaInput.Delete();
```

Transferbereiche suchen

Um den Transferbereich mit Positionsnummer und erweiterter Positionsnummer abzurufen, verwenden Sie folgenden Programmcode:

```
// Find a transfer area with Positionnummer and ExtendedPositionNumber  
TransferArea transferAreaFind = transferAreas.Find(4, 3);
```

Hinweis

Die erweiterte Positionsnummer wird nur für Transferbereiche mit mehreren Submodulen wie MSI benötigt.

Um den ersten Transferbereich mit dieser Positionsnummer abzurufen, verwenden Sie folgenden Programmcode:

```
// Find a transfer area with Positionnummer  
TransferArea transferAreaFind = transferAreas.Find(5);
```

Hinweis

Wenn eines dieser Attribute nicht verfügbar ist, kommt es zu einer Ausnahme.

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.8.37 Virtuelle Openness-Module/Submodule für ET 200SP PN HF

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness virtuelle Module/Submodule für ET 200 SP PN HF hinzufügen und löschen.

Die folgenden Eigenschaften werden bei virtuellen Modulen unterstützt:

Name der Eigenschaft	Datentyp	EomAtomValue	EomAtom-Beschreibung
SharedDeviceAccessConfigured	Boolean		
Name	String		
PositionNumber	Int32		
VirtualType	UInt64	1	MsoLocal
AddSubModules	Int64		

Die folgenden Eigenschaften werden bei virtuellen Submodulen unterstützt:

Eigenschaft	Datentyp	EomAtomValue	EomAtom-Beschreibung
SharedDeviceAccessConfigured	Boolean		
Comment	String		
PositionNumber	Int32		
Name	String		
ActivateDataStatus	Boolean		
InputAddressLength	Int64		
OutputAddressLength	Int64		
VirtualSubType	UInt64	1	MsoLocal
		2	MsoSub

Programmcode: Virtuelles Modul erstellen und löschen

Ändern Sie folgenden Programmcode, um ein neues Modul hinzuzufügen:

```
string Type = "OrderNumber:6ES7 155-6AU30-0CN0/V4.2";
Device ET200SP = new Project.Devices.CreateWithItem(Type, "ET200SP", "ET200SP");
DeviceItem Rack = ET200SP.DeviceItems.First();
string TypeIdentifier = "OrderNumber:ET200SP.Virtual.Module";
string Name = "VirtualIO_1";
int PositionNumber = 100;
DeviceItem VIM = Rack.PlugNew(TypeIdentifier, Name, PositionNumber);
```

Ändern Sie den folgenden Programmcode, um ein Modul zu löschen:

```
DeviceItem Rack = ET200SP.DeviceItems.First();  
string TypeIdentifier = "OrderNumber:ET200SP.Virtual.Module";  
string Name = "VirtualIO_1";  
int PositionNumber = 100;  
DeviceItem VIM = Rack.PlugNew(TypeIdentifier, Name, PositionNumber);  
VIM.Delete();
```

Programmcode: Virtuelles Submodul erstellen und löschen

Ändern Sie folgenden Programmcode, um in Openness virtuelle Submodule zu erstellen:

```
VIM1.SetAttribute("AddSubModules", (Int64)1);  
VIM2.SetAttribute("AddSubModules", (Int64)2);
```

Hinweis

Um ein Submodul hinzuzufügen, müssen Sie die Moduleigenschaft "AddSubModules" auf die Anzahl der hinzugefügten Submodule setzen. Wenn Sie versuchen, mehr Submodule als zulässig hinzuzufügen, wird eine Ausnahme ausgegeben.

Ändern Sie folgenden Programmcode, um ein Submodul zu löschen:

```
SubModule3.Delete();
```

Hinweis

Die ersten zwei Submodule können Sie nicht löschen.

5.8.38 Zugriff auf Domänen-Einstellungen

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness auf die vollständige Domänenverwaltung zugreifen. Die beiden folgenden neuen Dienste werden der Klasse Subnet hinzugefügt, damit auf die Domänenverwaltung über TIA Portal Openness zugegriffen werden kann:

- Dienst MrpDomainOwner
- Dienst SyncDomainOwner

Programmcode: Auf MrpDomainOwner zugreifen

Der Dienst MrpDomain liefert dem Navigator MrpDomains, was die MrpDomainComposition enthält. Die Zusammensetzung enthält Objekte vom Typ MrpDomain.

```
Subnet subnet = ...;
MrpDomainOwner mrpDomainOwner = subnet.GetService<MrpDomainOwner>();
MrpDomainComposition mrpDomainComposition = mrpDomainOwner.MrpDomains;
```

Ändern Sie folgenden Programmcode, um eine neue Domäne namens newMrpDomain zu erstellen:

```
int countOfMrpDomains = mrpDomainComposition.Count;
MrpDomain someMrpDomain = mrpDomainComposition.ElementAt(3);
MrpDomain firstMrpDomain = mrpDomainComposition.First();
MrpDomain byName = mrpDomainComposition.Find("DomainName");
// create a new domain
MrpDomain newMrpDomain = mrpDomainOwner.MrpDomains.Create("newMrpDomain");
```

Ändern Sie folgenden Programmcode, um Attributwerte einer MRP-Domäne zu lesen/zuschreiben:

```
NetworkInterface toBeAdded = ...;
newMrpDomain.SetAttribute("IsDefault", true);
newMrpDomain.SetAttribute("ManagerOutsideOfProjectActive", false);
var participants = firstMrpDomain.DomainParticipants;
int count = participants.Count;
participants.Add(toBeAdded);
foreach (NetworkInterface networkIf in participants)
{
    // do something at the interface
}
newMrpDomain.Delete();
```

Programmcode: Auf SyncDomainOwner zugreifen

Der Dienst stellt die SyncDomains des Navigators bereit, die die SyncDomainComposition enthalten. Diese Zusammensetzung enthält Objekte vom Typ SyncDomain.

```
Subnet subnet = ...;
SyncDomainOwner syncDomainOwner = subnet.GetService<SyncDomainOwner>();
SyncDomainComposition syncDomainComposition = syncDomainOwner.SyncDomains;
```

Ändern Sie folgenden Programmcode, um eine neue SyncDomain mit einem bestimmten Namen zu erstellen

```
int countOfSyncDomains = syncDomainComposition.Count;
SyncDomain someSyncDomain = syncDomainComposition.ElementAt(3);
SyncDomain firstSyncDomain = syncDomainComposition.First();
SyncDomain byName = syncDomainComposition.Find("DomainName");
// create a new domain
SyncDomain newSyncDomain = syncDomainOwner.SyncDomains.Create("newSyncDomain");
```

Ändern Sie folgenden Programmcode, um den Attributwert von SyncDomain zu lesen/zuschreiben:

```
NetworkInterface toBeAdded = ...;
string convertedName = newSyncDomain.ConvertedName;
newSyncDomain.SetAttribute("IsDefault", true);
newSyncDomain.SetAttribute("HighPerformanceActive", true);
newSyncDomain.SetAttribute("FastForwardingActive", true);
var participants = firstSyncDomain.DomainParticipants;
int count = participants.Count;
participants.Add(toBeAdded);
foreach (NetworkInterface networkIf in participants)
{
    // do something at the interface
}
newSyncDomain.Delete();
```

Siehe auch

[Verbindung zum TIA Portal aufbauen \(Seite 84\)](#)

[Projekt öffnen \(Seite 131\)](#)

5.8.39 Auf Attribut Redundanzbetrieb zugreifen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness auf ein S2-fähiges Gerät zugreifen, das als Switched S1-Gerät betrieben werden soll. Mit Openness sind Sie in der Lage, die Einstellung für 'Modus' in der TIA Portal Benutzeroberfläche für ein redundanzfähiges IOD von S2 auf 'S1' zu ändern, wie im folgenden Beispiel:

Network overview				
Connections				
Relations				
I/O communication				
VPN				
Telecontrol				
Offline configuration Online assignment				
Partner 1	Partner 2	Interface partner 2	Mode	
1 PLC_1				
2 PROFINET interface_1				
3 X1	IO device_5	PROFINET interface	IO device(S2)	
4 PLC_2				
5 PROFINET interface_1				
6 X1	IO device_5	PROFINET interface	IO device(S2)	
7				

Die Eigenschaft 'Modus' von TIA Portal steht in TIA Portal Openness mit dem Namen RedundancyMode zur Verfügung. Dieses Attribut ist für die PROFINET-Schnittstelle von S2- und R1-IO-Geräten verfügbar.

Die folgende Tabelle zeigt die Verwendung dieses Attributs.

Attribut	Beschreibung
NetworkInterface.GetAttribute("RedundancyMode")	Lesen des Attributs über Openness
NetworkInterface.SetAttribute("RedundancyMode", "RedunancyMode.S2")	Einstellen des Attributs über Openness
NetworkInterface.SetAttribute("RedundancyMode", "RedunancyMode.SwitchedS1")	

Programmcode

Ändern Sie den folgenden Programmcode, um auf das Redundanzmodus-Attribut zuzugreifen:

```
Device S2device = ...;
string s_S2DeviceInterfacelName = "";
NetworkInterface networkInterfacel = S2device.DeviceItems.First(f => f.Name ==
s_S2DeviceInterfacelName).GetService<NetworkInterface>();
string s_RedundancyMode = "...";
networkInterfacel.SetAttribute(s_RedundancyMode, RedundancyMode.SwitchedS1);
//Assert
Assert.AreEqual(RedundancyMode.SwitchedS1,
networkInterfacel.GetAttribute(s_RedundancyMode);
//Act
var systemBlockGroups = plc1?.GetService<ICompilable>();
CompilerResult compilerResult = systemBlockGroups?.Compile();
//Assert
Assert.AreEqual(0, compilerResult?.ErrorCount);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.8.40 CCDX-Transferbereich erstellen

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit der Überladung Create() des MulticastableTransferArea-Objekts in TIA Portal Openness einen CCDX-Transferbereich erstellen. Hierbei bezieht sich "multicastable" bzw. Multicastfähigkeit auf eine Eins-zu-viele-Beziehung, bei der ein Sendetransferbereich Informationen an mehrere Partner-Empfangstransferbereiche übertragen kann. In Gegenrichtung kann ein Empfangstransferbereich dagegen nur mit einem Sendepartner kommunizieren.

Methode

Die Überladung Create() akzeptiert in Openness die folgenden Parameternamen:

Parameter	Rückgabebetyp	Beschreibung
Create(NetworkInterface, TransferAreaType)	MulticastableTransferArea	Erstellt einen Transferbereich und fügt diesen einer vorhandenen CCDX-Verbindung mit dem angegebenen Typ (z. B. DDX) an der angegebenen Netzwerkschnittstelle hinzu
Create(NetworkInterface, TransferAreaType, Name)	MulticastableTransferArea	Erstellt einen Transferbereich unter dem angegebenen Namen und fügt den Bereich einer vorhandenen CCDX-Verbindung mit dem angegebenen Typ (z. B. DDX) an der angegebenen Netzwerkschnittstelle hinzu
Create(NetworkInterface, TransferAreaType, Name, Length)	MulticastableTransferArea	Erstellt einen Transferbereich mit dem angegebenen Namen und der angegebenen Länge und fügt den Bereich einer vorhandenen CCDX-Verbindung mit dem angegebenen Typ (z. B. DDX) an der angegebenen Netzwerkschnittstelle hinzu.
Create(MulticastableTransferArea, TransferAreaType)	MulticastableTransferArea	Erstellt einen neuen Transferbereich mit dem angegebenen Typ und weist den Bereich als Empfänger einem vorhandenen Sendetransferbereich hinzu.

Programmcode

```
// Given the NetworkInterfaces
private void CreatingTransferArea()
{
    NetworkInterface senderNetworkInterface = ...;
    NetworkInterface receiverNetworkInterface = ...;
    MulticastableTransferArea transferArea1 =
    senderNetworkInterface.MulticastableTransferAreas.Create(receiverNetworkInterface,
    TransferAreaType.DDX);
    string name = "Example_CCDX_TA";
    MulticastableTransferArea transferArea2 =
    senderNetworkInterface.MulticastableTransferAreas.Create(receiverNetworkInterface,
    TransferAreaType.DDX, name);
    int length = 32;
    MulticastableTransferArea transferArea3 =
    senderNetworkInterface.MulticastableTransferAreas.Create(receiverNetworkInterface,
    TransferAreaType.DDX, name, length);
    // Create receiver only
    MulticastableTransferArea transferArea = ...;
    MulticastableTransferArea transferArea4 =
    senderNetworkInterface.MulticastableTransferAreas.Create(transferArea,
    TransferAreaType.DDX);
}
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.8.41 CCDX-Transferbereich löschen

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit der Überladung Delete() des MulticastableTransferArea -Objekts in TIA Portal Openness den CCDX-Transferbereich löschen. Hierbei bezieht sich "multicastable" bzw. Multicastfähigkeit auf eine Eins-zu-viele-Beziehung, bei der ein Sendetransferbereich Informationen an mehrere Partner-Empfangstransferbereiche übertragen kann. In Gegenrichtung kann ein Empfangstransferbereich dagegen nur mit einem Sendepartner kommunizieren.

Beim Löschen eines Empfängers hingegen verhält sich die Überladung Delete() anders. Es gibt immer nur einen Sender (Eins-zu-viele-Beziehung).

Sender	Empfänger	Verhalten
1	c = 1	Sowohl der Sender als auch der Empfänger werden gelöscht
1	c > 1	Nur der Empfänger wird gelöscht

Folgendes Verhalten ist zu erwarten, wenn der Sender gelöscht wird:

Sender	Empfänger	Verhalten
1	Beliebige	Sowohl der Sender als auch alle Empfänger werden gelöscht

Programmcode

```
private void DeleteTransferArea()
{
    // Given the selected NetworkInterface
    NetworkInterface networkInterface = ...;
    // The name of the TransferArea
    string transferAreaName = "Example_CCDX_TA";
    networkInterface.MulticastableTransferAreas.FirstOrDefault(item => item.Name ==
transferAreaName).Delete();
}
```

Siehe auch

Projekt öffnen (Seite 131)

5.8.42 Konfiguration von CCDX-Transferbereichen ändern

Voraussetzung

- TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Konfiguration von CCDX-Transferbereichen ändern. In Openness können Sie über NetworkInterface als TransferAreaMulticastableComposition-Objekte, die alle Transferbereiche derselben Schnittstelle derselben PLC enthalten, auf Transferbereiche zugreifen.

Hierbei bezieht sich "multicastable" bzw. Multicastfähigkeit auf eine Eins-zu-viele-Beziehung, bei der ein Sendetransferbereich Informationen an mehrere Partner-Empfangstransferbereiche übertragen kann. In der Gegenrichtung kann ein Empfangstransferbereich jedoch nur mit einem Sendepartner (z B. dem CCDX-Transferbereich) kommunizieren.

Eigenschaft

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Address	AddressComposition	Ruft Adressinformationen wie Anfangsadresse, Länge, Io-Type usw. eines Transferbereichs vom Typ Siemens.Engineering.HW.Address ab	Lesen/Schreiben
Parent	IEngineeringObject	Ruft das übergeordnete EOM-Element eines Transferbereichs ab	Schreibgeschützt
Comment	String	Ruft die Eigenschaft Comment eines Transferbereichs ab	Lesen/Schreiben
Direction	TransferAreaDirection	Ruft die Eigenschaft Direction eines Transferbereichs vom Typ TransferAreaDirection ab. Gibt an, dass die Kommunikationsrichtung davon abhängt, ob der Transferbereich Sender oder Empfänger ist.	Schreibgeschützt
DataLength	Int	Ruft die Datenlänge (z.B. Adresslänge) eines Transferbereichs ab bzw. stellt diese ein	Lesen/Schreiben
Name	String	Ruft die Eigenschaft Name eines Transferbereichs ab	Lesen/Schreiben

5.8 Funktionen für Netzwerke

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
PartnerTransferAreas	MulticastableTransferAreaAssociation	Ruft die Partner-Transferbereiche eines multicastfähigen Transferbereichs für zum Beispiel CCDX ab Beispielsweise enthält der Sendetransferbereich einen oder mehrere Empfangstransferbereiche auf einem oder mehreren Partnergeräten. Ein Sendetransferbereich kann mehrere Empfangspartner haben. Sowohl ein Sende- als auch ein Empfangstransferbereich müssen mindestens einen Partner haben; sie können nicht einzeln vorhanden sein. Daher müssen beim Löschen eines Sendetransferbereichs auch alle zugehörigen Empfangspartner gelöscht werden. Wenn weitere Empfangspartner vorhanden sind, hat das Löschen eines Empfangstransferbereichs dagegen nicht zwangsläufig die Löschung seines Sendepartners zur Folge.	Schreibgeschützt
Type	TransferAreaType	Ruft die Eigenschaft Type eines Transferbereichs vom Typ TransferAreaType ab. Ein multicastfähiger Transferbereich kann folgenden Enum-Wert aufweisen: TransferAreaType.DDX wobei sich DDX auf den Transferbereichstyp 'CCDX' für die Kommunikation über direkten Datenaustausch bezieht.	Schreibgeschützt
MulticastableTransferAreas	MulticastableTransferAreaComposition	Ruft die Eigenschaft MulticastableTransferAreas der Netzwerkschnittstelle ab. Enthält multicastfähige Transferbereiche (z. B für CCDX-Transferbereiche) einer Netzwerkschnittstelle. Dabei kann es sich sowohl um Sende- als auch um Empfangstransferbereiche handeln.	Schreibgeschützt

Attribut	Datentyp	Beschreibung	Zugriff
TransferUpdateTime	Int32	Aktualisierungszeit für die Übertragung des Providerelements (Sender)	Schreibgeschützt

Programmcode

```
private void ModifyingCCDXTransferArea()
{
    // Given the selected NetworkInterface
    NetworkInterface networkInterface = ...;
    // Navigate to the desired MulticastableTransferArea
    string transferAreaName = "Example_CCDX_TA";
    MulticastableTransferArea transferArea =
    networkInterface.MulticastableTransferAreas.FirstOrDefault(item => item.Name ==
    transferAreaName);
    // Properties
    Property properties = transferArea.Properties;
    // Address
    Address address = transferArea.Address;
    // Parent
    IEngineeringObject parent = transferArea.Parent;
    // Comment
    string comment = transferArea.Comment;
    // Direction
    TransferAreaDirection direction = transferArea.Direction;

    // Name
    string name = transferArea.Name;
    // PartnerTransferAreas
    MulticastableTransferAreaAssociation partnerTransferAreas =
    transferArea.PartnerTransferAreas;
    // TransferUpdateTime
    int transferUpdateTime = (int)transferArea.GetAttribute("TransferUpdateTime");
    // Type
    TransferAreaType type = transferArea.Type;
    // DataLength
    transferArea.DataLength = 12;
}
```

5.9 Funktionen auf Geräte

5.9.1 Obligatorische Attribute von Geräten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Jedes Gerät oder Geräteelement besitzt bestimmte obligatorische Attribute, die gelesen und/oder geschrieben werden können. Diese Attribute sind immer die gleichen wie in der Benutzeroberfläche des TIA Portals.

In Openness werden die folgenden Attribute unterstützt:

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
Author	String	read/write	dynamisch	
Comment	String	read/write	dynamisch	manchmal schreibgeschützter Zugriff
CommentML	MultilingualTextItem	read/write	dynamisch	manchmal schreibgeschützter Zugriff
IsGsd	Bool	read		WAHR, wenn die Gerätebeschreibung über GSD/GSDML installiert wird
Name	String	read/write		manchmal schreibgeschützter Zugriff
Typeldentifier	String	read		
TypeName	String	read	dynamisch	

Programmcode: Obligatorische Attribute eines Geräts

Ändern Sie den folgenden Programmcode, um die obligatorischen Attribute eines Geräts abzurufen:

```
Device device = ...;
string nameValue = device.Name;
bool isGsdValue = device.IsGsd;
```

Programmcode: Obligatorische Attribute bei dynamischem Zugriff

Ändern Sie den folgenden Programmcode, um die Attribute bei dynamischem Zugriff aufzurufen:

```
Device device = ...;
var attributeNames = new[]
{
    "TypeName", "Author", "Comment";
foreach (var attributeName in attributeNames){object attributeValue =
    ((IEngineeringObject)device).GetAttribute(attributeName);
}
```

5.9.2 Typkennung von Geräten und Geräteelementen abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Das Attribut `TypeIdentifier` wird zur Identifizierung eines Hardwareobjekts verwendet, das über TIA Portal Openness API erstellt werden kann. `TypeIdentifier` ist eine Zeichenkette, die aus mehreren Teilen besteht: `<TypeIdentifierType>:<Identifier>`

Mögliche Werte für `TypeIdentifierType` sind:

- `OrderNumber`
- `GSD`
- `System`

OrderNumber

`OrderNumber` ist die allgemeine `TypeIdentifier` für alle im Hardware-Katalog vorhandenen Module.

Format der Typkennung	Beispiel	Besonderheiten
<code><OrderNumber></code>	OrderNumber:3RK1 200-0CE00-0AA2	-
<code><OrderNumber>/<FirmwareVersion></code>	OrderNumber:6ES7 510-1DJ01-0AB0/ V2.0	Die Firmware-Version ist optional, wenn diese gar nicht oder nur eine Version im System vorhanden ist. Bitte beachten
<code><OrderNumber>// <AdditionalTypeIdentifier></code>	OrderNumber:6AV2 124-2DC01-0AX0// Landscape	Die zusätzliche Typkennung kann notwendig sein, wenn <code>OrderNumber</code> und <code>FirmwareVersion</code> keinen eindeutigen Treffer im System haben.

Hinweis

Es gibt einige wenige Module im Hardware-Katalog, die Platzhalterzeichen in der Bestellnummer enthalten, welche eine bestimmte Gruppe realer Hardware repräsentiert, z. B. die verschiedenen Längen der S7-300 Baugruppenträger. In diesem Fall können sowohl die `OrderNumber` als auch die Platzhalter-`OrderNumber` verwendet werden, um eine Instanz des Hardwareobjekts zu erstellen. Platzhalter können jedoch nicht beliebig an irgendeiner Stelle verwendet werden.

GSD

Dies ist die Kennung, die für Module verwendet wird, die über GSD oder GSDML dem TIA Portal hinzugefügt werden.

Format der Typkennung	Beispiel	Besonderheiten
<GsdName>/<GsdType>	GSD:SIEM8139.GSD/DAP	GsdName ist der Name des GSD oder GSDML in Großbuchstaben. GsdType ist eines der folgenden: • D: Gerät (device) • R: Baugruppenträger (rack) • DAP: Kopfmodul • M: Modul • SM: Submodul GsdId ist die Typkennung.
<GsdName>/<GsdType>/<GsdId>	GSD:SIEM8139.GSD/M/4	

System

Dies ist die Kennung für Objekte, die nicht mittels `OrderNumber` oder GSD bestimmt werden können.

Format der Typkennung	Beispiel	Besonderheiten
<SystemTypeIdentifier>	System:Device.S7300	SystemTypeIdentifier ist die primäre Kennung eines Objekts. AdditionalTypeIdentifier kann notwendig sein, wenn die SystemTypeIdentifier nicht eindeutig ist. Die Präfixe für bestimmte Objekttypen sind: • Connection. • Subnet. • Device. • Rack.
<SystemTypeIdentifier>/<AdditionalTypeIdentifier>	GSD:SIEM8139.GSD/M/4	

Programmcode

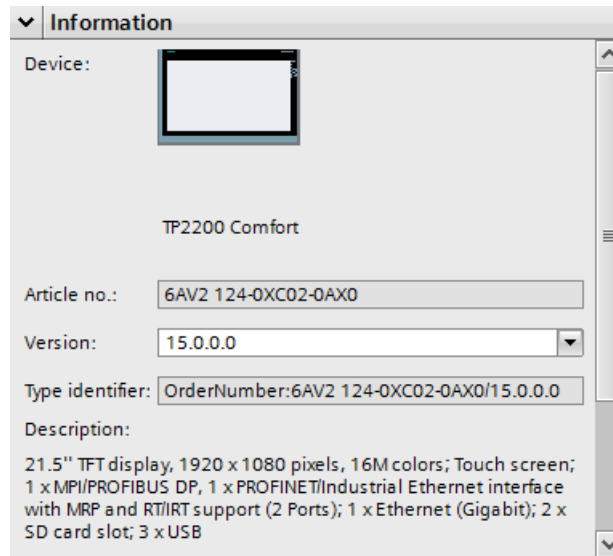
Um die Typkennung für vom Benutzer verwaltbare und getrennt erstellbare Objekte für GSD abzurufen, ändern Sie den folgenden Programmcode:

```
HardwareObject hardwareObject = ...;
string typeIdentifier = hardwareObject.TypeIdentifier;
```


Typkennungen im TIA Portal anzeigen

Wenn Sie eine Typkennung kennen müssen, ermitteln Sie sie im TIA Portal wie folgt:

1. Aktivieren Sie unter "Optionen > Einstellungen > Hardware-Konfiguration > Anzeige der Typkennung" die Einstellung "Anzeige der Typkennung für Geräte und Module".
2. Öffnen Sie den Editor "Geräte & Netze".
3. Wählen Sie ein Gerät im Katalog aus.
Die Typkennung wird unter "Information" angezeigt.



5.9.3 App-ID in Gerät und Geräteelementen einstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Anwendungs-ID im Gerät und in den Geräteelementen eines TIA Portal-Projekts einstellen, damit diese Anwendungs-IDs in der aktuellen Sitzung des TIA Portal zur späteren Verwendung gespeichert werden.

Sie stellen die Anwendungs-ID ein, indem Sie der API das Paar aus "Anwendungsschlüssel" und "Anwendungswert" bereitstellen.

Es gelten folgende Randbedingungen für die Anwendungs-ID aus Geräte- und Geräteelementobjekt:

- Ein Objekt eines Typs, der diese Funktion unterstützt, darf maximal 64 verfügbare Anwendungs-IDs haben.
- Die Schlüssel- und/oder Wertlänge der Anwendungs-ID darf maximal 128 Zeichen umfassen.
- Der Schlüssel der Anwendungs-ID eines bestimmten Objekts ist eindeutig.
- Es darf nur eine Anwendungs-ID mit einem "Anwendungsschlüssel" eingestellt werden.
- Wenn ein anderer Anwendungswert mit dem gleichen "Anwendungsschlüssel" eingestellt wird, wird der zuletzt eingestellte Anwendungswert gespeichert.

Programmcode

Ändern Sie folgenden Programmcode, um die Anwendungs-ID für Geräte- und Geräteelementobjekt einzustellen:

```
Device device = ...;  
// Ask for Service CustomIdentityProvider on the device/device item  
var customIdentityProviderService = device.GetService<CustomIdentityProvider>();  
//Set the Application ID (Key-Value) pair  
customIdentityProviderService.Set("Application_Key", "Application_Value");
```

5.9.4 App-ID in Gerät und Geräteelementen abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Anwendungs-ID im Gerät und in den Geräteelementen eines TIA Portal-Projekts abrufen, wenn für das Gerät oder die Geräteelemente bereits eine Anwendungs-ID eingestellt ist.

Anwendungs-IDs werden als Paar aus "Anwendungsschlüssel" und "Anwendungswert" als Bestandteil des TIA Portal gespeichert. Sie können mit TIA Portal Openness den Wert der Anwendungs-ID durch Eingabe des "Anwendungsschlüssels" abrufen.

Programmcode

Ändern Sie folgenden Programmcode, um den Wert der Anwendungs-ID für ein Objekt abzurufen, das bereits mit dem Anwendungsschlüssel eingestellt wurde:

```
Device device = ...;  
// Ask for Service CustomIdentityProvider on the device/device item  
var customIdentityProviderService = device.GetService<CustomIdentityProvider>();  
customIdentityProviderService.Set("Application_Key", "Application_Value");  
//Get the Application ID (Value) for the given Application Key  
var applicationValue = customIdentityProviderService.Get("Application_Key");
```

Siehe auch

Projekt öffnen (Seite 131)

5.9.5 App-ID in Gerät und Geräteelementen entfernen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Anwendungs-ID (benutzerdefinierte Identität) im Gerät und in den Geräteelementen eines TIA Portal-Projekts entfernen, damit das Gerät und die Geräteelemente mit der relevanten Anwendungs-ID aktualisiert werden können.

Eine CustomIdentityNotFoundException wird zurückgegeben, wenn die Anwendungs-ID, die als Argument übergeben wurde, nicht vorhanden ist.

Programmcode

Ändern Sie folgenden Programmcode, um die Anwendungs-ID für Gerät und Geräteelemente zu entfernen:

```
Device device = ...;
// Ask for Service CustomIdentityProvider on the device/device item
var customIdentityProviderService = device.GetService<CustomIdentityProvider>();
//Remove the CustomIdentity corresponding to Application ID
try
{
    customIdentityProviderService.Remove("Application_Key");
}
catch (CustomIdentityNotFoundException ex)
{
    // When CustomIdentity is not found for a the given key "Application_Key".
}
```

5.9.6 Erstellen eines Geräts

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Ein Gerät kann auf zwei Arten erstellt werden, in einem Projekt oder in einer Gerätegruppe:

- Erstellen eines Geräts über eine Geräteelement-Typkennung wie im TIA-Hardwarekatalog
`Device CreateWithItem(DeviceItemId, DeviceItemName, DeviceName)`
- Nur Gerät erstellen
`Device Create(DeviceTypeId, DeviceName)`

Name	Typ	Beschreibung
DeviceItemId	String	Typkennung des Geräteelements
DeviceTypeId	String	Typkennung des Geräts
DeviceItemName	String	Name des erstellten Geräteelements
DeviceName	String	Name des erstellten Geräts

Siehe: Typkennung von Geräten und Geräteelementen abrufen (Seite 347)

Programmcode: Gerät mit Typkennung erstellen

Um ein Geräteobjekt über eine Typkennung zu erstellen, ändern Sie den folgenden Code:

```
DeviceComposition devices = ...;
Device device = devices.CreateWithItem("OrderNumber:6ES7 510-1DJ01-0AB0/V2.0", "PLC_1",
    "NewDevice");
Device gsdDevice = devices.CreateWithItem("GSD:SIEM8139.GSD/M/4 ", "GSD Module",
    "NewGsdDevice");
```

Programmcode: Nur Gerät erstellen

Um nur das Geräteobjekt zu erstellen, ändern Sie den folgenden Code:

```
DeviceComposition devices = ...;
Device deviceOnly = devices.Create("System:Device.S7300", "S7300Device");
Device gsdDeviceOnly = devices.Create("GSD:SIEM8139.GSD/D", "GSD Device");
```

5.9.7 Geräte enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Die TIA Portal Openness API ordnet Geräte ähnlich der Projektnavigation im TIA Portal an. PNV.

- Geräte, die dem Projekt direkt untergeordnet sind, werden in der Zusammensetzung "Devices" des Projekts gesammelt angeordnet.
- Geräte, die sich in Geräteordnern befinden, werden in der Zusammensetzung "Devices" des Ordners gesammelt angeordnet.

Hinweis

Hierarchie von Hardware-Objekten des Objektmodells (Seite 67) beachten.

Um die Geräte eines Projekts zu enumerieren, verwenden Sie eine der folgenden Möglichkeiten:

- Alle Geräte auf erster Ebene enumerieren
- Alle Geräte in Gruppen oder Untergruppen enumerieren

- Alle Geräte eines keine Gerätegruppen enthaltenden Projekts enumerieren
- Alle Geräte ungruppierter Gerätesystemgruppen enumerieren

Beispiele für enumerierbare Geräte:

- Central station
- PB-Slave / PN-IO device
- HMI Device

Programmcode: Geräte auf erster Ebene enumerieren

Um Geräte auf erster Ebene zu enumerieren, ändern Sie folgenden Programmcode:

```
private static void EnumerateDevicesInProject(Project project)
{
    DeviceComposition deviceComposition = project.Devices;
    foreach (Device device in deviceComposition)
    {
        // add code here
    }
}
```

Ändern Sie den folgenden Programmcode, um auf ein einzelnes Gerät zuzugreifen:

```
private static void AccessSingleDeviceByName(Project project)
{
    DeviceComposition deviceComposition = project.Devices;
    // The parameter specifies the name of the device
    Device device = deviceComposition.Find("MyDevice");
}
```

Programmcode: Geräte in Gruppen oder Untergruppen enumerieren

Um auf Geräte in einer Gruppe zuzugreifen, müssen Sie zuerst zu der Gruppe und danach zu dem Gerät navigieren.

```
//Enumerate devices in groups or sub-groups
private static void EnumerateDevicesInGroups(Project project)
{
    foreach (DeviceUserGroup deviceUserGroup in project.DeviceGroups)
    {
        EnumerateDeviceUserGroup(deviceUserGroup);
    }
}
private static void EnumerateDeviceUserGroup(DeviceUserGroup deviceUserGroup)
{
    EnumerateDeviceObjects(deviceUserGroup.Devices);
    foreach (deviceUserGroup subDeviceUserGroup in deviceUserGroup.Groups)
    {
        // recursion
        EnumerateDeviceUserGroup(subDeviceUserGroup);
    }
}
private static void EnumerateDeviceObjects(DeviceComposition deviceComposition)
{
    foreach (Device device in deviceComposition)
    {
        // add code here
    }
}
```

Programmcode: Spezifische Geräte finden

Zum Finden eines bestimmten Geräts unter seinem Namen ändern Sie folgenden Programmcode:

```
//Find a specific device by name
Project project = ...
Device plc1 = project.Devices.First(d => d.Name == "Mydevice");
... // Work with the device
```

Zum Finden eines bestimmten Geräts über die Methode "Find" ändern Sie folgenden Programmcode:

```
//Find a specific device via "Find" method
Project project = ...
Device plc1 = project.Devices.Find("MyDevice");
... // Work with the device
```

Programmcode: Geräte eines keine Gerätegruppen enthaltenden Projekts enumerieren

Ändern Sie folgenden Programmcode:

```
//Enumerate all devices which are located directly under a project that contains no device groups
Project project = ...
foreach (Device device in project.Devices)
{
    ... // Work with the devices
}
```

Programmcode: Alle Geräte in einem Ordner enumerieren

Ändern Sie folgenden Programmcode:

```
//Enumerate all devices located in a folder
Project project = ...
DeviceUserGroup sortingGroup = project.DeviceGroups.Find ("Sorting");
Device plc1 = sortingGroup.Devices.First(d => d.Name == "MyStationName");
... // Work with the device
```

Programmcode: Geräte ungruppierter Gerätesystemgruppen enumerieren

Zum Strukturieren der Projekte wurden dezentrale Geräte in die Gruppe UngroupedDevices eingefügt. Um auf diese Gruppe zuzugreifen, navigieren Sie zuerst zur Gruppe und danach zum Gerät.

Ändern Sie folgenden Programmcode:

```
//Enumerate devices of the ungrouped device system group
Project project = ...
DeviceSystemGroup group = project.UngroupedDevicesGroup;
Device plc1 = group.Devices.First(d => d.Name == "MyStationName");
... // Work with the device
```

5.9.8 Auf Geräte zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Jedes GSD- oder GSDML-basierte Gerät verfügt über Attribute. Einige davon dienen zur Ermittlung des genauen Typs des Geräts.

Name	Datentyp	Schreibbar	Zugriff	Beschreibung
Author	String	read/write	dynamisch	-
Comment	String	read/write	dynamisch	-
GsdName	String	read	dynamisch	Name der GSD- oder GSDML-Datei.
GsdType	String	read	dynamisch	Typ des Hardwareobjekts. Der Wert des Geräts ist stets "D".
GsdId	String	read	dynamisch	Spezifische Kennung des Hardwareobjekts. Für Geräte immer leer.
IsGsd	Bool	read	modelliert	WAHR für GSD-Gerät oder GSDML-Gerät
Name	String	read/write	modelliert	-
TypeIdentifier	String	read	modelliert	-

Programmcode: Identifikationsattribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId"
};
foreach (var attributeName in attributeNames) {
    object attributeValue = device.GetAttribute(attributeName);
}
```

Programmcode: Attribute

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Device device = ...;
string nameValue = device.Name;
bool isGsdValue = device.IsGsd;
```

Programmcode: Attribute bei dynamischem Zugriff

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId"
};
foreach (var attributeName in attributeNames) {
    object attributeValue = device.GetAttribute(attributeName);
}
```

Programmcode: Besonderheiten von GSD-Geräten

Ein Gerät, das zugleich ein GSD-Gerät ist, bietet zusätzliche Funktionalität. Um die Funktion GsdDevice abzurufen, verwenden Sie die Methode GetService.

```
GsdDevice gsdDevice = ((IEngineeringServiceProvider)deviceItem).GetService<GsdDevice>();
if (gsdDevice != null) {
    ... // work with the GSD device
};
```

Programmcode: Attribute eines GSD-Geräts

Programmcode: Attribute eines GSD-Geräts

```
Device device = ...;
GsdDevice gsdDevice = ...;
string gsdId = gsdDevice.GsdId;
string gsdName = gsdDevice.GsdName;
string gsdType = gsdDevice.GsdType;
bool isProfibus = gsdDevice.IsProfibus;
bool isProfinet = gsdDevice.IsProfinet;
```

5.9.9 Auf den TIA Portal-Hardwarekatalog zugreifen**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können erforderliche Hardware im Katalog manuell recherchieren, bevor Sie die Openness-Anwendung ausführen. Sie können mit Hilfe von TIA Portal Openness auch korrekt auf Benutzeranfragen reagieren. Beispielsweise können Sie, wenn ein Gerät, das zur Openness-Anwendung hinzugefügt werden sollte, im Hardwarekatalog nicht vorhanden ist, dem Benutzer eine Nachricht zukommen lassen, dass die GSD(ML)-Datei oder das HSP zuerst installiert werden muss.

Attribut

Ein Siemens.Engineering.HW.HardwareCatalog.CatalogEntry neuen Typs mit den folgenden Attributmitgliedern wird eingeführt:

Attributname	Datentyp	Beschreibung	Zugriff
ArticleNumber	System.String	Gibt die Artikelnummer des Hardwareobjekts an, zum Beispiel: 6ES7 516-3AN00-0AB0	Lesen
CatalogPath	System.String	Gibt den Katalogpfad des Artikels an, zum Beispiel: Root\PLC\SIMATIC S7-1500\CPU\CPU 1516-3 PN/ DPI6ES7 516-3AN00-0AB0	Lesen

Attributname	Datentyp	Beschreibung	Zugriff
Description	System.String	Gibt die Beschreibung des Eintrags im Hardwarekatalog an, zum Beispiel: • Zentralbaugruppe mit Anzeige, Arbeitsspeicher 1 MB für Programm und 5 MB für Daten • Bitbearbeitungszeit 10 ns • Vierstufiges Schutzkonzept, integrierte Technologiefunktionen: Motion Control, Regeln, Zählen und Messen • Integrierte Trace-Funktion • 1. Schnittstelle: PROFINET IO Controller, unterstützt RT/IRT, zwei Ports, MRP, Transportprotokoll TCP/IP, S7-Kommunikation, Webserver, konstante Buszykluszeit, Routing • 2. Schnittstelle: PROFINET-Basisdienste, Transportprotokoll TCP/IP-Protokoll, Webserver, Routing • 3. Schnittstelle: PROFI-BUS DP Master, konstante Buszykluszeit, Routing; Firmware V1.0	Lesen
Typidentifizier	System.String	Gibt die Typkennung eines Artikels an, zum Beispiel: Bestellnummer: 6ES7 516-3AN00-0AB0/V1.0	Lesen
TypidentifizierNormalized	System.String	Gibt die normalisierte Typkennung des Artikels an, zum Beispiel: Bestellnummer 6ES7516-3AN00-0AB0/V1.0	Lesen
TypeName	System.String	Gibt den Typnamen des Geräts an, zum Beispiel CPU 1516-3 PN/DP	Lesen
Version	System.String	Gibt die Version des Eintrags im Hardwarekatalog an, zum Beispiel: V1.0	Lesen

Programmcode

Die Funktion mit einem Parameter, die mit einem leeren String gefiltert wird, gibt den vollständigen verfügbaren Hardwarekatalog zurück:

```
TiaPortal portal = TiaPortal.GetProcesses().First().Attach();
var itemList = portal.HardwareCatalog.Find(string.Empty);
```

Die Funktion mit einem Parameter ermittelt Hardwarekatalog-Einträge und kann mit einem Filterstring gefiltert werden:

```
TiaPortal portal = TiaPortal.GetProcesses().First().Attach();
var itemList = portal.HardwareCatalog.Find("1516");
var first = itemList.First();
Console.WriteLine(first.TypeIdentifier);
//proj.Devices.Create("OrderNumber:6ES7 516-2GN00-0AB0/V2.9", "PLC_1");
Console.ReadKey();
```

Die Funktion mit zwei Parametern filtert das Suchergebnis auf kompatible/steckbare Geräte und wird außerdem auf einen angegebenen TypeIdentifier gefiltert:

```
TiaPortal portal = TiaPortal.GetProcesses().First().Attach();
var itemList = portal.HardwareCatalog.Find("1BL00", "OrderNumber:6ES7 516-2GN00-0AB0/V2.9");
//The itemList contains all the devices which are compatible/pluggable with the given
//TypeIdentifier
//and filtered by the filter string. In this case one of the itemList elements is 6ES7
//521-1BL00-0AB0/V1.0.
Console.ReadKey();
```

5.9.10 Löschen eines Geräts

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie den folgenden Programmcode, um ein Gerät zu löschen:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
namespace DeletingDevice
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            Project project = ...;
            // Needs to be: UngroupedDevicesGroup
            Device deviceToDelete = project.UngroupedDevices.Devices.Find(".....");
            // delete device
            deviceToDelete.Delete();
        }
    }
}
```

5.9.11 Massenänderung von Hardware-Parametern

Voraussetzung

- TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können TIA Portal Openness zur Konfiguration verschiedener Attribute eines Hardware-Objekts mit einem einzigen SetAttributes-Aufruf verwenden.

API beseitigt auch das Problem mit der Reihenfolge und den Abhängigkeiten der Attribute und ihrer entsprechenden Werte, wenn die Funktion aufgerufen wird.

Signatur

SetAttributes(IEnumerable<KeyValuePair<string, object>> attributes, AttributeDelegate errorHandler);

Hinweis

Dies ist eine Überlastung der SetAttributes (IEnumerable<KeyValuePair<String, Objekt>>-Attribute) API.

Ab TIA Portal V19 funktioniert die Massenänderung von Hardware-Parametern mit Reihenfolge bei allen überlasteten "SetAttributes"-Methoden gleich.

Programmcode

```
using Siemens.Engineering;
...
{
    TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
    var project = tiaPortal.Projects.Open(new FileInfo(project));
    IList<KeyValuePair<string, object>> attributesToSet = new List<KeyValuePair<string,
    object>>
    {
        new KeyValuePair<string, object>("IsochronousTi", (double)0.1), new KeyValuePair<string,
        object>("IsochronousTo", (double)0.3),
        new KeyValuePair<string, object>("IsochronousTiToCalculationMode",
        Siemens.Engineering.HW.IsochronousTiToCalculationMode.Manual),
        new KeyValuePair<string, object>("IsochronousMode", true),
    };
    project.DeviceItems[0].SetAttributes(attributesToSet, MyErrorHandler));
    private void MyErrorHandler(AttributeConfiguration attributeConfiguration)
    {
        //this means, if one of the attribute set fails, the mechanism won't stop
        //just skip the problematic part and the other attributes will be set.
        attributeConfiguration.CurrentSelection = AttributeChoiceSelection.Ignore;
    }
}
```

5.10 Funktionen auf Geräteelemente

5.10.1 Obligatorische Attribute von Geräteelementen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Jedes Gerät oder Geräteelement besitzt bestimmte obligatorische Attribute, die gelesen und/oder geschrieben werden können. Diese Attribute sind immer die gleichen wie in der Benutzeroberfläche des TIA Portals.

In TIA Portal Openness werden die folgenden Attribute unterstützt:

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
Author	string	read/write	dynamisch	-
Classification	DeviceItemClassification	read	modelliert	Zur Klassifikation des Geräteelements, Enum-Flags, siehe Beschreibung der Klassifikation der Geräteelemente.
Comment	string	read/write	dynamisch	manchmal schreibgeschützter Zugriff
CommentML	MultilingualTextItem	read/write	dynamisch	manchmal schreibgeschützter Zugriff
FirmwareVersion	string	read	dynamisch	-
InterfaceOperatingMode	InterfaceOperatingModes	read/write	dynamisch	Für Geräteelemente mit der Funktion <code>NetworkInterface</code>
InterfaceType	NetType	readwrite	dynamisch	Für Geräteelemente mit der Funktion <code>NetworkInterface</code>
IsBuiltIn	bool	read	modelliert	FALSCH für Objekte, die durch den Anwender erstellt werden können
IsGsd	bool	read	modelliert	WAHR, wenn die Gerätebeschreibung über GSD/GSDML installiert wird
IsPlugged	bool	read	modelliert	WAHR für gesteckte Geräte
Label	string	read	dynamisch	Für Geräteelemente mit der Funktion <code>NetworkPort</code> oder <code>NetworkInterface</code> . Wenn die Schnittstelle oder der Port kein "Label" hat, dann ist <code>Label</code> gleich <code>String.Empty</code> .
LocationIdentifier	string	read/write	dynamisch	-

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
Name	string	read/write	modelliert	manchmal schreibgeschützter Zugriff
OrderNumber	string	read	dynamisch	-
PlantDesignation	string	read/write	dynamisch	-
PositionNumber	int	read	modelliert	-
TypeIdentifier	string	read	modelliert	-
TypeName	string	read	dynamisch	Der sprachenunabhängige Typname. Optional für Geräteelemente, die vom Anwender als automatisch-erstellte Elemente oder feste Submodule nicht verwaltbar sind).
InstallationDate	DateTime	read/write	dynamisch	-
AdditionalInformation	string	read/write	dynamisch	-

Klassifikation des Geräteelements

Wert	Beschreibung
Siemens.Engineering.HW.DeviceItemClassifications.None	Keine Klassifikation.
Siemens.Engineering.HW.DeviceItemClassifications.CPU	Das Geräteelement ist eine CPU
Siemens.Engineering.HW.DeviceItemClassifications.HM	Das Geräteelement ist ein Kopfmodul.

Programmcode: Obligatorische Attribute eines Geräteelements

Ändern Sie den folgenden Programmcode, um die obligatorischen Attribute eines Geräteelements abzurufen:

```
DeviceItem deviceItem = ...;
string nameValue = deviceItem.Name;
string typeIdentifierValue = deviceItem.TypeIdentifier;
int positionNumberValue = deviceItem.PositionNumber;
bool isBuiltInValue = deviceItem.IsBuiltIn;
bool isPluggedValue = deviceItem.IsPlugged;
```

Programmcode: Obligatorische Attribute bei dynamischem Zugriff

Ändern Sie den folgenden Programmcode, um die Attribute bei dynamischem Zugriff aufzurufen:

```
Device device = ...;
var attributeNames = new[] {
    "TypeName", "Author", "Comment", "OrderNumber", "FirmwareVersion", "PlantDesignation",
    "LocationIdentifier"
};
foreach (var attributeName in attributeNames) {
    object attributeValue = ((IEngineeringObject)deviceItem).GetAttribute(attributeName);
}

DeviceItem deviceItem = ...;
((IEngineeringObject)deviceItem).SetAttribute("Comment", "This is a comment.");
```

5.10.2 Ein Geräteelement erstellen und stecken**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Die Aktion `PlugNew(string typeIdentifizier, string name, int positionNumber)` von `HardwareObject` wird verwendet, um

- ein neues Geräteelement zu erstellen und in ein vorhandenes Hardwareobjekt zu stecken
- ein neues Untergeräteelement zu erstellen, z. B. ein Submodul, und dieses in ein Geräteelement zu stecken

Wenn die Aktion erfolgreich war, gibt sie das erstellte Geräteelementobjekt aus. Andernfalls wird eine wiederherstellbare Ausnahme ausgelöst.

Mit der Aktion `CanPlugNew(string typeIdentifizier, string name, int positionNumber)` können Sie ermitteln, ob Erstellen und Stecken möglich sind. Wenn eine Ausführung nicht möglich ist, gibt die Aktion `false` aus.

Wenn die Methode den Wert `True` zurückgibt, kann die Aktion aus den folgenden unvorhersehbaren Gründen dennoch fehlschlagen.

- eine Positionsnummer wird bereits von einem anderen Geräteelement verwendet
- das aktuelle Geräteelement kann nicht in die Position gesteckt werden, obwohl sie frei ist
- der Behälter stellt die Positionsnummer nicht bereit

- der Name des Geräteelements wird bereits von einem vorhandenen Geräteelement im selben Behälter verwendet
- das Geräteelement kann nicht in den Behälter gesteckt werden
- das Gerät ist online

Die folgende Tabelle zeigt die benötigten Methodenparameter:

Name	Typ	Beschreibung
typeIdentifier	String	Typkennung des erstellten Geräteelements
name	String	Name des erstellten Geräteelements
positionNumber	int	Positionsnummer des erstellten Geräteelements

Programmcode

Um ein Geräteelement in ein vorhandenes Hardwareobjekt zu stecken, ändern Sie den folgenden Programmcode:

```
HardwareObject hwObject = ...;
string typeIdentifier = ...;
string name = ...;
int positionNumber = ...;
if(hwObject.CanPlugNew(typeIdentifier, name, positionNumber))
{
    DeviceItem newPluggedDeviceItem = hwObject.PlugNew(typeIdentifier, name,
positionNumber);
}
```

Auf Modulinformationen zugreifen

Der TIA Portal Openness-Anwender kann über das Objekt ModuleInformationProvider auf Informationen zu den steckbaren Modulen zugreifen. Der Anwender hat Zugriff auf

- die Behältertypen, in die ein bestimmtes Modul zu stecken ist (z. B. Gerät und Baugruppenträger) – über die Methode FindContainerTypes.
- die verfügbaren Versionen für ein bestimmtes teilweise angegebenes Modul – über die Methode FindModuleTypes.

Programmcode: Auf das Objekt ModuleInformationProvider zugreifen

```
Project project = ...;
HardwareUtilityComposition extensions = project.HwUtilities;
var result = extensions.Find("ModuleInformationProvider") as ModuleInformationProvider;
```

Programmcode: Mit der Methode FindContainerTypes auf Behältertypen zugreifen

Die Methode FindContainerTypes gibt die Behältertypen eines angegebenen Moduls zurück. Das Modul wird über den Parameter typeIdentifier angegeben. Die resultierende Liste enthält die Typidentifiers sämtlicher Behältertypen des angeforderten Typs. Üblicherweise umfasst dies ein Gerät und einen Baugruppenträger, und die Behälter werden in ihrer Reihenfolge in der Hierarchie im Projekt mit Beginn am Gerät angegeben.

Name des Parameters	Typ	Beschreibung
typeIdentifier	String	Typkennung eines Geräteelements

ACHTUNG

Diese Methode funktioniert nur bei die in der Netzsicht sichtbaren Modulen.

Diese Methode funktioniert nur bei Modulen, nicht bei Submodulen.

```
string typeIdentifier = ...;
string[] containerTypes = moduleInformationProvider.FindContainerTypes(typeIdentifier);
```

Programmcode: Mit der Methode FindModuleTypes auf Versionen zugreifen

Die Methode FindModuleTypes gibt alle möglichen Versionen eines Hardwareobjekts unter Verwendung der teilweisen Typkennung des Geräteelements zurück. Diese Methode gibt eine Liste von Strings aus. Jeder String entspricht dem vollständigen Typidentifizier einer möglichen Übereinstimmung für den teilweisen Typidentifizier.

Eine gültige teilweise Typkennung darf nur vollständige Teile enthalten, wobei jeder Teil in der Typkennung durch das Zeichen "/" getrennt ist. Platzhalter oder unvollständige Teile werden nicht unterstützt. Außerdem müssen die folgenden Einschränkungen im Hinblick auf die Mindestanzahl von angegebenen Teilen beachtet werden:

- Bestellnummer mindestens ein Teil. Beispiel: OrderNumber:6ES7 317-2EK14-0AB0
- GSD: mindestens zwei Teile. Beispiel: GSD:SI05816A.GSD/M
- System: mindestens ein Teil. Beispiel: System:Rack.ET200SP

Name des Parameters	Typ	Beschreibung
partialTypeIdentifier	String	teilweise Typkennung eines Geräteelements

```
string partialTypeIdentifier = ...;
string[] moduleTypes = moduleInformationProvider.FindModuleTypes(partialTypeIdentifier);
```

Programmcode: Mit der Methode GetPlugLocations auf die Steckorte zugreifen

Die Methode GetPlugLocations gibt die Informationen über Steckplätze wie Steckort, Positionsnummer (Bezeichnung eines Steckplatzes) und verfügbare Steckplätze für das Hardwareobjekt zurück.

Die Klasse PlugLocation hat die folgenden Eigenschaften.

Name der Eigenschaft	Typ	Beschreibung
PositionNumber	Int	Die Positionsnummer des freien Steckplatzes
Label	String	Die Bezeichnung des freien Steckplatzes

- Ist für eine bestimmte Positionsnummer kein "label" vorhanden, wird die String-Darstellung der Positionsnummer verwendet.
- PlugLocation-Objekte werden nur für freie Steckplätze bereitgestellt.

```
HardwareObject hardwareObject = ...;
IList<PlugLocation> result = hardwareObject.GetPlugLocations();
foreach (PlugLocation item in result)
{
    Console.WriteLine("{0} - {1}", item.PositionNumber, item.Label);
}
```

5.10.3 Geräteelemente in einen anderen Steckplatz verschieben**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit der Aktion PlugMove(DeviceItem deviceItem, int positionNumber) von HardwareObject kann ein vorhandenes Geräteelement verschoben und in ein vorhandenes Hardwareobjekt gesteckt werden. Die Methode PlugMove fügt die Geräteelemente ein, bei denen das Modul die Bedienoberfläche nicht stecken konnte. In diesen Fällen wird die Aktion PlugMove mit Übersetzungsfehlern beendet.

Die Aktion CanPlugMove(DeviceItem deviceItem, int positionNumber) dient zum Ermitteln der Bewegungsmöglichkeit. Wenn die Bewegung nicht möglich ist, gibt CanPlugMove den

Wert False zurück. Wenn die Methode den Wert True zurückgibt, kann die Aktion aus den folgenden unvorhersehbaren Gründen dennoch fehlschlagen.

- eine Positionsnummer wird bereits von einem anderen Geräteelement verwendet
- das aktuelle Geräteelement kann nicht in die Position gesteckt werden, obwohl sie frei ist
- der Behälter stellt die Positionsnummer nicht bereit
- der Name des Geräteelements wird bereits von einem vorhandenen Geräteelement im selben Behälter verwendet
- das Geräteelement kann nicht in den Behälter gesteckt werden
- das Geräteelement kann vom Benutzer nicht gesteckt werden
- das Geräteelement kann vom Benutzer nicht entfernt werden
- das Gerät ist online

Programmcode

Ändern Sie folgenden Programmcode:

```
HardwareObject hwObject = ...;
DeviceItem deviceItemToMove = ...;
int positionNumber = ...;
if(hwObject.CanPlugMove(deviceItemToMove, positionNumber)
{
DeviceItem movedDeviceItem = hwObject.PlugMove(deviceItemToMove, positionNumber);
}
```

5.10.4 Geräteelement kopieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Verwenden Sie die Aktion `PlugCopy(DeviceItem deviceItem, int positionNumber)` von `HardwareObject`, um ein Gerät innerhalb eines Projekts zu kopieren und in vorhandene Hardware zu stecken. In seltenen Fällen kann die Methode `PlugCopy` dann funktionieren, wenn eine Baugruppe nicht in die UI gesteckt werden kann. In diesem Fall treten nach dem Kopieren Übersetzungsfehler auf. Wenn `PlugCopy` erfolgreich war, wird die Kopie des Geräteelementobjekts zurückgegeben. Andernfalls wird eine wiederherstellbare Ausnahme ausgelöst.

Mögliche Ursachen für eine fehlgeschlagene Aktion:

- eine Positionsnummer wird bereits von einem anderen Geräteelement verwendet
- das aktuelle Geräteelement kann nicht in die Position gesteckt werden, obwohl sie frei ist
- der Behälter stellt die Positionsnummer nicht bereit
- der Name des Geräteelements wird bereits von einem vorhandenen Geräteelement im selben Behälter verwendet
- das Geräteelement kann nicht in den Behälter gesteckt werden
- das Geräteelement kann nicht in die UI gesteckt werden
- ...

Mit der Aktion CanPlugCopy(DeviceItem deviceItem, int positionNumber) können Sie bestimmen, ob der Kopiervorgang möglich ist. Wenn der Kopiervorgang nicht ausgeführt werden kann, gibt CanPlugCopy den Wert False zurück. Wenn die Methode jedoch den Wert True zurückgibt, kann die Aktion aus unvorhersehbaren Gründen dennoch fehlschlagen.

Name des Parameters	Typ	Beschreibung
deviceItem	DeviceItem	Zu kopierendes Geräteelement
positionNumber	Int	Positionsnummer zum Kopieren des Geräteelements

Programmcode

Ändern Sie folgenden Programmcode:

```
HardwareObject hwObject = ...;
DeviceItem deviceItemToCopy = ...;
int positionNumber = ...;
if(hwObject.CanPlugCopy(deviceItemToCopy, positionNumber))
{
    DeviceItem copiedDeviceItem = hwObject.PlugCopy(deviceItemToCopy, positionNumber);
}
```

5.10.5 Einstellungen für benutzerdefiniertes Logo vornehmen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Einstellungen für das benutzerdefinierte Logo im CPU-Display der S7-1500 PLC vornehmen. Die Funktion wird nicht von allen S7-1500 PLCs unterstützt.

Die Funktion FrontPanel Display ist am Geräteelement CPU-Display verfügbar und stellt folgende Methode bereit:

Methodenname	Datentyp	Beschreibung
SetUserDefinedLogo()	FileInfo	Festlegen des Pfads für das Bild, das in das CPU-Display geladen werden soll. Das SetUserDefinedLogo() kann nur verwendet werden, wenn die Eigenschaft the UserDefinedLogoActivated aktiviert ist. Andernfalls wird eine Ausnahme EngineeringTargetInvocation ausgelöst, wenn die Eigenschaft UserDefinedLogoActivated deaktiviert ist.

Folgende Eigenschaften sind in Openness nur verfügbar, wenn die CPU ein Display umfasst und die Eigenschaften auch im TIA Portal verfügbar sind.

Name der Eigenschaft	Datentyp	Beschreibung
UserDefinedLogoActivated	boolean	Zeigt das benutzerdefinierte Logo an. Um das Bild in das CPU-Display laden zu können, muss die Eigenschaft UserDefinedLogoActivated aktiviert sein. Das Bild kann nicht geladen werden, wenn die Eigenschaft UserDefinedLogoActivated deaktiviert ist. Andernfalls müssen Sie die anderen Einstellungen nach der Deaktivierung der Eigenschaft UserDefinedLogoActivated zurücksetzen.
AdaptLogoActivated	boolean	Zeigt das angepasste Logo an. Die Eigenschaft AdaptLogoActivated ist nur verfügbar, wenn UserDefinedLogoActivated aktiviert ist.

Programmcode

```
using Siemens.Engineering.HW.Features.FrontPanelDisplay;
private void UserDefinedLogoSettings()
{
    FrontPanelDisplay frontPanelDisplay = plcDisplay.GetService<FrontPanelDisplay>();

    //Enabling the user defined logo in the display
    frontPanelDisplay.UserDefinedLogoActivated = true;

    //Enabling the adapt logo in the display
    frontPanelDisplay.AdaptLogoActivated = true;

    //Uploading an image file in the display
    frontPanelDisplay.SetUserDefinedLogo(new FileInfo(@"C:\Pictures\CompanyLogo.jpg"));
}
```


5.10.6 Geräteelement löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um ein Geräteelement zu löschen, ändern Sie folgenden Programmcode:

```
Project project = ...;
var device = project.UngroupedDevicesGroup.Devices.Find(".....");
var deviceItem = deviceItem.DeviceItems.First();
// delete device item
deviceItem.Delete();
```

5.10.7 Geräteelemente enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Um ein Geräteelement abzurufen, verwenden Sie das HardwareObject Die Elemente eines Hardwareobjekts sind die in das Hardwareobjekt gesteckten Teile, die der Anwender im TIA Portal sehen kann:

- ein Baugruppenträger in einem Gerät
- ein Modul in einem Baugruppenträger
- Ein Submodul in einem Modul
- Ein Submodul in einem Submodul

Hinweis

Weiterführende Informationen zu diesem Thema finden Sie im Kapitel Hierarchie von Hardware-Objekten des Objektmodells (Seite 67).

Programmcode: Geräteelement eines Geräts enumerieren

Ändern Sie den folgenden Programmcode, um Geräteelemente eines Hardware-Objekts zu enumerieren:

```
private static void EnumerateDeviceItems (HardwareObject hardwareObject)
{
    foreach (DeviceItem deviceItem in hardwareObject.Items)
    {
        // add code here
    }
}
```

Programmcode: Mit Zusammensetzungshierarchie enumerieren

Ändern Sie den folgenden Programmcode, wenn Sie die Geräteelemente eines Geräts mit Hilfe der Zusammensetzungshierarchie enumerieren möchten:

```
//Enumerates devices using an composition
private static void EnumerateDeviceItems (Device device)
{
    DeviceItemComposition deviceItemComposition = device.DeviceItems;
    foreach (DeviceItem deviceItem in deviceItemComposition)
    {
        // add code here
    }
}
```

Programmcode: Geräteelemente mit Zuordnung enumerieren

Ändern Sie den folgenden Programmcode, um die Geräteelemente mit einer Zuordnung zu enumerieren:

```
//Enumerates devices using an association
private static void EnumerateDeviceItemsWithAssociation (Device device)
{
    DeviceItemAssociation deviceItemAssociation = device.Items;
    foreach (DeviceItem deviceItem in deviceItemAssociation)
    {
        // add code here
    }
}
```

5.10.8 Geräteelemente aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Zum Aufrufen von Objekten vom Typ "DeviceItem" verwenden Sie die folgenden Attribute:

- Name (string): Name des Geräteelements
- Behälter (HardwareObject): Behälter, in den das Geräteelement gesteckt wird

Name	Datentyp	Schreibbar	Zugriff	Beschreibung
Author	string	read/write	dynamisch	-
Comment	string	read/write	dynamisch	-
FirmwareVersion	string	read	dynamisch	Nur für Kopfmodule
GsdName	string	read	dynamisch	Name der GSD-Datei.
GsdType	string	read	dynamisch	Typ des Hardwareobjekts. Der Wert des Geräts ist stets "D".
GsdId	string	read	dynamisch	Spezifische Kennung des Hardwareobjekts. Für Geräte immer leer.
IsBuiltIn	bool	read	-	-
IsGsd	bool	read	modelliert	WAHR für GSD-Gerät oder GSDML-Gerät
IsPlugged	bool	read	modelliert	-
IsProfibus	bool	read	modelliert	-
IsProfinet	bool	read	modelliert	-
Name	string	read/write	modelliert	-
OrderNumber	string	read	dynamisch	Nur für Kopfmodule
PositionNumber	bool	read	modelliert	-
TypeIdentifier	string	read	modelliert	-

Programmcode: Geräteelement aufrufen

Um ein Geräteelement aufzurufen, ändern Sie folgenden Programmcode:

```
public static DeviceItem AccessDeviceItemFromDevice(Device device)
{
    DeviceItem deviceItem = device.DeviceItems[0];
    return deviceItem;
}
```

Programmcode: Auf Geräteelement von einem Geräteelement zugreifen

Ändern Sie folgenden Programmcode, um auf ein Geräteelement eines Geräteelements zuzugreifen:

```
public static DeviceItem AccessDeviceItemFromDeviceItem(DeviceItem deviceItem)
{
    DeviceItem subDeviceItem = deviceItem.DeviceItems[0];
    return subDeviceItem;
}
```

Programmcode: Zum Behälter eines Geräteelements navigieren

Ändern Sie folgenden Programmcode über das Attribut "Container" von DeviceItem, um zurück zum Behälter eines Geräteelements zu navigieren:

```
DeviceItem deviceItem = ...;
HardwareObject container = deviceItem.Container;
```

Programmcode: Identifikationsattribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId" };
foreach (var attributeName in attributeNames) {
    object attributeValue = ((IEngineeringObject)deviceItem).GetAttribute(attributeName);
}
```

Programmcode: Attribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
GsdDeviceItem gsdDeviceItem =
    ((IEngineeringServiceProvider)deviceItem).GetService<GsdDeviceItem>();

string gsdName = gsdDeviceItem.GsdName;
string gsdType = gsdDeviceItem.GsdType;
string gsdId = gsdDeviceItem.GsdId;
bool isProfinet = gsdDeviceItem.IsProfinet;
bool isProfibus = gsdDeviceItem.IsProfibus;;
```

Programmcode: Attribute bei dynamischem Zugriff abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
GsdDeviceItem gsdDeviceItem =
    ((IEngineeringServiceProvider)deviceItem).GetService<GsdDeviceItem>();

var attributeNames = new[] {
    "TypeName", "Author", "Comment", ...
};
foreach (var attributeName in attributeNames) {
    object attributeValue =
        ((IEngineeringObject)gsdDeviceItem).GetAttribute(attributeName);
}
```

Programmcode: Attribute festlegen

Um die Attribute festzulegen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
((IEngineeringObject)deviceItem).SetAttribute("Comment", "This is a
comment.");
```

Programmcode: prm-Daten eines Kopfmoduls abrufen

```
DeviceItem deviceItem = ...;
GsdDeviceItem gsdDeviceItem =
    ((IEngineeringServiceProvider)deviceItem).GetService<GsdDeviceItem>();

int dsNumber = 0;           // For Profibus GSDs, dataset number zero must be used!
int byteOffset = 0;
int lengthInBytes = 5;

// read complete data set:
byte[] prmDataComplete = gsdDeviceItem.GetPrmData(dsNumber, byteOffset, lengthInBytes);

// read partial data set (only second byte):
byteOffset = 1;
lengthInBytes = 1;
byte[] prmDataPartial = gsdDeviceItem.GetPrmData(dsNumber, byteOffset, lengthInBytes);
```

Programmcode: prm-Daten eines Kopfmoduls einstellen

Um die prm-Daten abzurufen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
GsdDeviceItem gsdDeviceItem =
    ((IEngineeringServiceProvider)deviceItem).GetService<GsdDeviceItem>();

// The parameters byteOffset and the length of the byte array prmData define the range
// within the
// dataset which is written to.
// For Profibus GSDs, dataset number zero must be used!

// Change the highlighted bytes 2-4 from 0x0 to 0x1
// to write only the first two bytes: byte[] prmData = {0x05, 0x21};

int dsNumber = 0;
int byteOffset = 0;
byte[] prmData = {0x05, 0x21, 0x01, 0x01, 0x01};

gsdDeviceItem.SetPrmData(dsNumber, byteOffset, prmData);
```

Siehe auch

Hierarchie von Hardware-Objekten des Objektmodells (Seite 67)

5.10.9 Auf das Geräteelement als Schnittstelle zugreifen**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Wenn ein Geräteelement eine Schnittstelle ist, bietet es gegenüber einem einfachen Geräteelement zusätzliche Funktionalität. Über diese Schnittstelle kann der Anwender auf die Knoten und die Betriebsart der Schnittstelle zugreifen. Aufgrund dieser Funktionalität kann das Geräteelement durch Zugriff auf die Funktion `NetworkInterface` als `IoDevice` (sekundär) oder als `IoController` (primär) verwendet werden (ein spezifischer Dienst des Geräteelements).

Auf die Eigenschaften der Schnittstelle wird über die Enumeration `InterfaceOperatingModes` zugegriffen.

Wert	Beschreibung
<code>InterfaceOperatingModes.None</code>	Standardeinstellung
<code>InterfaceOperatingModes.IoDevice</code>	Betriebsart der Schnittstelle "IoDevice" (Slave).

Wert	Beschreibung
InterfaceOperatingModes.IoController	Betriebsart der Schnittstelle "IoController" (Master).
InterfaceOperatingModes.IoDevice or InterfaceOperatingModes.IoController	Schnittstellenbetrieb: beide der oben genannten.

Programmcode: Auf die Netzwerkschnittstellenfunktion zugreifen

Um die Netzwerkschnittstellenfunktion abzurufen, ändern folgenden Programmcode:

```
Device deviceItem = ...;
NetworkInterface itf =
    ((IEngineeringServiceProvider)deviceItem).GetService<NetworkInterface>();
if (itf != null)
{
    //Working with the interface
}
//Accessing nodes and operating mode
NodeComposition nodes = itf.Nodes;
InterfaceOperatingModes mode = itf.InterfaceOperatingMode;
//Accessing the type of interface
NetType itfType = itf.InterfaceType;
//Modifying the operating mode and interface type
itf.InterfaceOperatingMode = InterfaceOperatingModes.IoDevice;
itf.InterfaceType = NetType.Profibus;
//Accessing the ports linked to an interface.
NetworkPortAssociation nodes2 = itf.Ports;
```

5.10.10 Auf "Gerät ändern" zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Mit der ChangeType()-API können Sie den Typ des Geräts oder Geräteelements ändern. Im Vergleich zur Benutzeroberfläche von TIA Portal gibt es bei dieser Funktion in TIA Portal Openness Einschränkungen:

- Es wird keine Liste mit möglichen Änderungspartnern bereitgestellt.
- Keine Rückmeldung zu den Typänderungen am geänderten Gerät oder Geräteelement.

Die in der Benutzeroberfläche von TIA Portal unterstützten Module können auch in TIA Portal Openness verwendet werden. Folgende Funktionalitäten werden bei der Funktion in TIA Portal Openness nicht unterstützt:

- GSDML-Austausch
- Aktualisierung der Modulbeschreibung

Bei folgenden Umständen kann die `ChangeType()`-API den Typ des Geräts oder Geräteelements nicht ändern:

- Der angegebene Typenidentifizierer ist unbekannt.
- Das Objekt, das geändert werden soll, ist schreibgeschützt.
- Die Typänderung wird nicht unterstützt, beispielsweise beim Wechsel zu einem vollkommen anderen Modultyp oder einer ganz anderen Modulfamilie.

Programmcode

```
using Siemens.Engineering.HW.DeviceItem;
private void ChangeTypeDeviceItem()
{
    DeviceItem rack = ...;
    DeviceItem deviceItem = rack.PlugNew("OrderNumber:6ES7 516-3AN01-0AB0/V2.8", "PLC_1", 1);
    try
    {
        deviceItem.ChangeType("OrderNumber:6ES7 516-3AN01-0AB0/V2.9");
    }
    catch (EngineeringTargetInvocationException e)
    {
        Console.WriteLine("ChangeType failed: {0}", e.Message);
    }
}
```

5.10.11 Auf Attribute einer I/O-Geräteschnittstelle zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Für Schreibzugriff ist der PLC offline.

Verwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Attribute für IRT und isochronen Modus an der I/O-Geräteschnittstelle abrufen oder festlegen.

Zugriff auf die Schnittstelle eines I/O-Controllers

Über die folgenden Attribute kann auf die Schnittstelle eines I/O-Controllers zugegriffen werden. Der Controller muss der Synchronisierungsmaster sein:

Attributname	Datentyp	Schreibbar	Zugriff	Beschreibung
PnSendClock	Int64	r/w	Dynamisches Attribut	Takt in Nanosekunden senden

Zugriff auf die Schnittstelle eines I/O-Systems

Über die folgenden Attribute kann auf die Schnittstelle eines I/O-Systems zugegriffen werden. Die Ti/To-Werte können von allen Modulen und Submodulen, die zum I/O-System gehören, verwendet werden.

Attributname	Datentyp	Schreibbar	Zugriff
IsochronousTiToAutoCalculation	BOOL	r/w	Dynamisches Attribut
IsochronousTi	DOUBLE	r/w	Dynamisches Attribut
IsochronousTo	DOUBLE	r/w	Dynamisches Attribut

Zugriff auf die Schnittstelle eines I/O-Geräts

Über die folgenden Attribute kann auf die Schnittstelle eines I/O-Geräts zugegriffen werden. Die Ti/To-Werte können von allen Modulen und Submodulen, die zum I/O-System gehören, verwendet werden.

Attributname	Datentyp	Schreibbar	Zugriff
IsochronousMode	BOOL	r/w	Dynamisches Attribut
IsochronousTiToCalculationMode	IsochronousTiToCalculationMode	r/w	Dynamisches Attribut
IsochronousTi	DOUBLE	r/w	Dynamisches Attribut
IsochronousTo	DOUBLE	r/w	Dynamisches Attribut

Die folgenden ENUM-Werte werden für das Attribut `IsochronousTiToCalculationMode` bereitgestellt:

Wert	Beschreibung
<code>IsochronousTiToCalculationMode.None</code>	-
<code>IsochronousTiToCalculationMode.FromOB</code>	Ti/To-Werte des OB (am IO-System konfiguriert) werden verwendet.
<code>IsochronousTiToCalculationMode.FromSubnet</code>	Dieser Wert wird von PROFINET-Schnittstellen nicht verwendet.
<code>IsochronousTiToCalculationModeAutomaticMinimum</code>	Ti/To-Werte werden für das IO-Device automatisch berechnet.
<code>IsochronousTiToCalculationMode.Manual</code>	Der Anwender kann für dieses IO-Device Ti/To-Werte manuell eingeben.

Programmcode: Attribute einer I/O-Geräteschnittstelle abrufen oder festlegen

Ändern Sie den folgenden Programmcode, um auf den Sendetaktwert zuzugreifen:

```
DeviceItem pnInterface = ...;
// read attribute
long attributeValue = (long)pnInterface.GetAttribute("PnSendClock");
// write attribute
long sendClock = 2000000;
pnInterface.SetAttribute("PnSendClock", sendClock);
```

Ändern Sie den folgenden Programmcode, um auf die Ti/To-Werte eines OB zuzugreifen:

```
IoSystem ioSystem = ...;
bool titoAutoCalculation = (bool)ioSystem.GetAttribute("IsochronousTiToAutoCalculation");
ioSystem.SetAttribute("IsochronousTiToAutoCalculation", true);
```

Ändern Sie den folgenden Programmcode, um auf die isochrone Einstellung einer I/O-Geräteschnittstelle zuzugreifen:

```
DeviceItem pnInterface = ...;
bool isochronousMode = (bool)pnInterface.GetAttribute("IsochronousMode");
pnInterface.SetAttribute("IsochronousMode", true);
```

5.10.12 Subnetz eines Geräteelements abrufen**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit dem Dienst NetworkInterface können Sie auf die Navigatorknoten zugreifen. Jeder Knoten bietet wiederum durch Aufrufen der Eigenschaft ConnectedSubnet Zugriff auf den Subnetzlink.

Programmcode

Ändern Sie folgenden Programmcode, um das Subnetz eines Geräts oder Geräteelements abzurufen:

```
using Siemens.Engineering.HW.Node;
private void GetSubnetDeviceItem()
{
    DeviceItem itfDeviceItem = ...;
    NetworkInterface itf = itfDeviceItem.GetService<NetworkInterface>();
    foreach (Node node in itf.Nodes)
    {
        Subnet subnet = node.ConnectedSubnet;
        // work with the subnet
    }
}
```

Hinweis

Bei einem nicht verbundenen Knoten ist das Subnetz im obigen Code Null.

5.10.13 Auf Attribute des IoController zugreifen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Für Schreibzugriff ist der PLC offline.

Anwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Attribute für den IoController abrufen oder festlegen. Die folgenden Attribute sind nur unter PROFINET IoController (unter einer PROFINET-Schnittstelle) verfügbar. Wenn der Anwender ein Attribut in der Benutzeroberfläche ändern kann, kann der Anwender das entsprechende Attribut auch über TIA Portal Openness ändern.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
SyncRole	SyncRole	Lesen-Schreiben	Dynamisches Attribut	-
PnDeviceNumber	int	Schreibgeschützt	Dynamisches Attribut	In der TIA Portal UI befindet sich diese Eigenschaft unter dem Knoten Ethernet (Abschnitt PROFINET)

Die Eigenschaft Synchronization role ist in der PROFINET-Schnittstelle der TIA Portal UI verfügbar. Die Enum SyncRole hat die folgenden Werte:

Enumerationswert	Numerischer Wert
SyncRole.NotSynchronized	0
SyncRole.SyncMaster	1
SyncRole.SyncSlave	2
SyncRole.RedundantSyncMaster	4

Programmcode: Attribute des IoController festlegen

```
IoController ioController= ...;
SyncRole syncRole = (SyncRole)((IEngineeringObject)ioController).GetAttribute("SyncRole");
((IEngineeringObject)ioController).SetAttribute("SyncRole", SyncRole.SyncMaster);
```

5.10.14 Auf Attribute des IoConnector zugreifen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Für Schreibzugriff ist der PLC offline.

Anwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Attribute für den IoConnector abrufen oder festlegen. Die folgenden Attribute sind nur unter PROFINET IoController (unter einer PROFINET-Schnittstelle) verfügbar. Wenn der Anwender ein Attribut in der Benutzeroberfläche ändern kann, kann der Anwender das entsprechende Attribut auch über TIA Portal Openness ändern.

Es gibt vier Typen von Attributen wie Attribute für die Aktualisierungszeit, Attribute für die Überwachungszeit, Attribute für die Synchronisation und Attribute für die Gerätenummer.

Attribute für die Aktualisierungszeit

Die Attribute für die Aktualisierungszeit werden nachstehend aufgeführt.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
PnUpdateTimeAutoCalculation	Boolean	Lesen-Schreiben	Dynamisches Attribut	Wenn dieses Attribut wahr ist, wird die Aktualisierungszeit automatisch berechnet.
PnUpdateTime	Int64	Lesen-Schreiben	Dynamisches Attribut	Die Aktualisierungszeit wird in Nanosekunden gemessen.
PnUpdateTimeAdaption	Boolean	Lesen-Schreiben	Dynamisches Attribut	-

Attribute für die Überwachungszeit

Die Attribute für die Überwachungszeit werden nachstehend aufgeführt.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
PnWatchdogFactor	Int32	Lesen-Schreiben	Dynamisches Attribut	-
PnWatchdogTime	Int64	Schreibgeschützt	Dynamisches Attribut	Die Überwachungszeit wird in Nanosekunden gemessen.

Attribute für die Synchronisation

Die Attribute für die Synchronisation werden nachstehend aufgeführt.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
RtClass	RtClass	Lesen-Schreiben	Dynamisches Attribut	-
SyncRole	SyncRole	Schreibgeschützt	Dynamisches Attribut	-

Die Enum RtClass hat folgende Werte.

Enumerationswert	Numerischer Wert
RtClass.None	0
RtClass.RT	1
RtClass.IRT	2

Die Enum SyncRole hat folgende Werte.

Enumerationswert	Numerischer Wert
SyncRole.NotSynchronized	0
SyncRole.SyncMaster	1
SyncRole.SyncSlave	2
SyncRole.RedundantSyncMaster	4

Attribute für die Gerätenummer

Die Attribute für die Gerätenummer werden nachstehend aufgeführt.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
PnDeviceNumber	int	Lesen-Schreiben	Dynamisches Attribut	Gibt die Gerätenummer an.

Programmcode: Attribute des IoConnector abrufen und festlegen

```
IoConnector connector = ...

var attributeNames = new[] {
    "PnUpdateTimeAutoCalculation", "PnUpdateTime", "PnUpdateTimeAdaption", "PnWatchdogFactor",
    "PnWatchdogTime", "RtClass", "SyncRole"
};

foreach (var attributeName in attributeNames)
{
    object attributeValue = ((IEngineeringObject)connector).GetAttribute(attributeName);
}

connector.SetAttribute("PnUpdateTimeAutoCalculation", true);
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.10.15 Auf einen Adresscontroller zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Ein Geräteelement, das zugleich ein Adresscontroller ist, bietet zusätzliche Funktionalität. Um auf die registrierten Adressen des Adresscontrollers zuzugreifen, muss die Rolle `AddressController` verwendet werden.

Programmcode: Adresscontroller abrufen

Um die Adresscontrollerrolle abzurufen, ändern Sie folgenden Programmcode:

```
AddressController addressController =  
(IEngineeringServiceProvider)deviceItem).GetService<AddressController>();  
if (addressController != null)  
{  
    ... // work with the address controller  
}
```

Attribute eines Adresscontrollers

Ein Adresscontroller hat folgende Attribute:

- `RegisteredAddresses`

Um die Attribute eines Adresscontrollers abzurufen, ändern Sie folgenden Programmcode:

```
AddressController addressController = ...;  
foreach (Address registeredAddress in addressController.RegisteredAddresses)  
{  
    ...  
}
```

5.10.16 Auf Adressen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Adressobjekte werden über den Zusammensetzungslink `Addresses` eines Geräteelements aufgerufen. Das Attribut `Addresses` gibt eine Sammlung von `AddressComposition` zurück, die enumeriert werden kann.

Programmcode: Adresse eines Geräteelements abrufen

Ändern Sie den folgenden Programmcode, um die Adresse eines Geräteelements abzurufen:

```
AddressComposition addresses = deviceItem.Addresses;
foreach(Address address in addresses)
{
    // work with the address
}
```

Programmcode: Adresse eines IO-Controllers abrufen

Ändern Sie den folgenden Programmcode, um die Adresse eines IO-Controllers abzurufen:

```
AddressComposition addresses = ioController.Addresses;
foreach(Address address in addresses)
{
    // work with the address
}
```

Attribute

Adresse unterstützt die folgenden Attribute:

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
AddressControllers	AddressControllerAssociation	read	-	-
Context	enum: AddressContext	read	dynamisch	nur für Diagnoseadressen und für spezifische Geräteelemente
IoType	enum: AddressIoType	read	-	-

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
StartAddress	Int32	read/write	modelliert	-
Length	Int32	read	-	-

Wert	Beschreibung
AddressIoType.Diagnosis	Der Adress-IO-Typ ist Diagnose.
AddressIoType.Input	Der Adress-IO-Typ ist Eingang.
AddressIoType.Output	Der Adress-IO-Typ ist Ausgang.
AddressIoType.Substitute	Der Adress-IO-Typ ist Ersatz.
AddressIoType.None	Der Adress-IO-Typ ist nicht angegeben.

Wert	Beschreibung
AddressContext.None	Der Adresskontext ist nicht gültig.
AddressContext.Device	Ein Geräte-Adressenkontext
AddressContext.Head	Ein Kopf-Adressenkontext

Programmcode: Attribute lesen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
AddressControllerAssociation addressControllers = address.AddressControllers;
Int32 startAddress = address.StartAddress;
AddressIoType addressType = address.IoType;
Int32 addressLength = address.Length;
```

Programmcode: Attribute schreiben

Um die Attribute zu schreiben, ändern Sie folgenden Programmcode:

```
Address addressControllers = ...;
address.StartAddress = intValueStartAddress;
```

Programmcode: Attribute bei dynamischem Zugriff

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Address address= ...;
object attributeValue = ((IEngineeringObject)address).GetAttribute("Context");
```

5.10.17 Auf "Hardwarekennung" zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Hardwarekennungsobjekte werden von den folgenden Objekten abgerufen:

- Device
- DeviceItem
- IoSystem

Die Hardwarekennung wird durch die Klasse `HwIdentifier` dargestellt und über das Attribut `HwIdentifiers` abgerufen.

Programmcode: Hardwarekennung abrufen

Um `HwIdentifier` verfügbar zu machen, ändern Sie folgenden Programmcode:

```
var hwObject = ...;
foreach (HwIdentifier hardwareIdentifier in hwObject.HwIdentifiers)
{
    // Work with the HwIdentifier
}
```

Attribute einer Hardwarekennung

```
HwIdentifierControllerAssociation controllers = hwIdentifier.HwIdentifierControllers;
Int64 Identifier = hwIdentifier.Identifier;
```

5.10.18 Auf Hardwarekennungscontroller zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Wenn ein Geräteelement gleichzeitig ein Hardwarekennungscontroller ist, kann auf die registrierten Hardwarekennungen zugegriffen werden. Um auf diese HwIdentifierController zuzugreifen, muss ein bestimmter Dienst des Geräteelements verwendet werden.

Programmcode: Hardwarekennungscontroller abrufen

Um den HwIdentifierController abzurufen, ändern Sie folgenden Programmcode:

```
HwIdentifierController hwIdentifierController =  
(IEngineeringServiceProvider)deviceItem).GetService<HwIdentifierController>();  
if (hwIdentifierController != null)  
{  
    ... // work with the hardware identifier controller  
}
```

Programmcode: Attribute eines Hardwarekennungscontrollers

Ein Adresscontroller hat folgende Attribute:

- **RegisteredHwIdentifiers:** Die Hardwarekennungscontroller, auf denen die Hardwarekennung registriert ist.

Um die Attribute eines Adresscontrollers abzurufen, ändern Sie folgenden Programmcode:

```
HwIdentifierController hwIdentifierController = ...;  
HwIdentifierAssociation controllers = hwIdentifierController.RegisteredHwIdentifiers;
```

5.10.19 Auf Kanäle von Geräteelementen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Ein Kanal wird durch die Klasse Channel dargestellt. Kanäle werden von einem Geräteelement über das Attribut Channels der Klasse DeviceItem abgerufen. Das Attribut Channels gibt eine Implementierung von ChannelComposition zurück, die enumeriert werden kann. Wenn das Geräteelement keine Kanäle hat, gibt das Attribut Channels eine leere Sammlung zurück.

Obligatorische Attribute

Ein Kanal unterstützt die folgenden obligatorischen Attribute:

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
IoType	ChannelIoType	read	modelliert	-
Type	ChannelType	read	modelliert	-
Number	Int32	read	modelliert	-
ChannelAddress	Int32	read	dynamisch	Adresse des Kanals in Bits
ChannelWidth	UInt32	read	dynamisch	Breite des Kanals in Bits

Programmcode: Kanäle von einem Geräteelement abrufen

Ändern Sie folgenden Programmcode, um die Kanäle eines Geräteelements abzurufen:

```
ChannelComposition channels = deviceItem.Channels
foreach(Channel channel in channels)
{
    // work with the channel
}
```

Programmcode: Obligatorische Attribute eines Kanals

Ändern Sie folgenden Programmcode, um die Kanäle eines Geräteelements abzurufen:

```
Channel channel = ...;
int channelNumber = channel.Number;
ChannelType type = channel.Type;
ChannelIoType ioType = channel.IoType;
```

Programmcode: Werte der Attribute bei dynamischem Zugriff abrufen

Um die Werte von dynamischen Attributen abzurufen, ändern Sie folgenden Programmcode:

```
Channel channel = ...;
Int32 channelAddress = (Int32)((IEngineeringObject)channel).GetAttribute("ChannelAddress");
UInt32 channelWidth = (UInt32)((IEngineeringObject)channel).GetAttribute("ChannelWidth");
```

Programmcode: Wert eines dynamischen Attributs festlegen

Um den Wert eines schreibbaren dynamischen Attributs festzulegen, ändern Sie folgenden Programmcode:

```
Channel channel = ...;
((IEngineeringObject)channel).SetAttribute("AnAttribute", 1234);
```

5.10.20 Auf normierte Typkennungen zugreifen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

In TIA Portal Openness können Sie mit `GetTypeIdentifierNormalized()` das normierte Format des Typidentifiers eines Moduls abrufen. Das Modul wird über einen Typidentifier angegeben.

`GetTypeIdentifierNormalized()` unterstützt folgende Eigenschaft:

Name der Eigenschaft	Datentyp	Beschreibung
typIdentifier	String	Gibt die Typkennung eines Geräteelements an

Programmcode

```
using Siemens.Engineering.HW.Utilities;  
private void ProvideTypeIdentifier(ModuleInformationProvider moduleInformationProvider)  
{  
    string typeIdentifier = ...;  
    string normalizedTypeIdentifier =  
        moduleInformationProvider.GetTypeIdentifierNormalized(typeIdentifier);  
}
```

5.10.21 PSC-Datei erstellen und exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen
Siehe Projekt öffnen (Seite 131)

Anwendung

Über die TIA Portal Openness API können Sie eine PSC-Datei erstellen und ein Gerät in die Datei laden. Um eine PSC-Datei zu erstellen und eine Gerätekonfiguration in die Datei zu laden, können Sie die Methode Export eines Dienstes CardReaderPscProvider verwenden. Der Dienst ist im Namensraum Siemens.Engineering.HW.Utilities vorhanden.

Programmcode

Ändern Sie folgenden Programmcode, um eine PSC-Datei zu erstellen und zu exportieren:

```
// preconditions
Device ipcDevice = myProject.Devices.Find("IPC427D");
FileInfo exportFile = new FileInfo(@"C:\Users\Ertan\Documents\Automation\PC system
configuration74.psc");
// get the card reader provider from hardware utilities
HardwareUtilityComposition utilities = myProject.HwUtilities;
HardwareUtility utility = utilities.Find("CardReaderPscProvider");
CardReaderPscProvider crp = (CardReaderPscProvider)utility;
// do the export
crp.Export(ipcDevice, exportFile);
```

Das Erstellen und Öffnen einer PSC-Datei mit separaten Befehlen wird nicht unterstützt. Wenn Sie eine bereits vorhandene PSC-Datei als Parameter angeben, wird die Datei nicht überschrieben und eine Ausnahme wird zurückgegeben. Ist die PSC-Datei noch nicht vorhanden, wird sie erstellt und der Ladevorgang durchgeführt.

Sie müssen sicherstellen, dass das Projekt problemlos übersetzt werden kann. Ansonsten wird eine Ausnahme zurückgegeben.

Hinweis

Aus Sicherheitsgründen wird der Exportvorgang für F-aktivierte Geräte in **Versionen V18 und früher** nicht unterstützt (beim Export wird eine Ausnahme zurückgegeben).

5.10.22 Verwalten von Verbindungen bei Erweiterungsbaugruppenträgern

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können mithilfe von TIA Portal Openness Verbindungen für Erweiterungsbaugruppenträger abrufen, hinzufügen und entfernen. Dadurch können Sie auch die Unterstützung von TIA Portal Openness für die Realisierung von Verbindungen für Erweiterungsbaugruppenträger beim CAX-Export/Import nutzen.

Programmcode

```
ImConnection imConnection = portDeviceItem.GetService<ImConnection>();  
imConnection.Connect(partnerport);  
imConnection.Disconnect();  
imConnection.GetPartnerPort();  
var imConnectionOwner = imConnection.OwnedBy;
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

5.11.1 Funktionen zum Laden von Daten ins PLC-Gerät

5.11.1.1 Zugriff auf Systemdiagnose-Eigenschaften von Plus-PLCs

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie Systemdiagnose-Eigenschaften für die Plus-PLC 1500 konfigurieren.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

In TIA Portal Openness werden die folgenden Systemdiagnose-Eigenschaften von PLC 1500 unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung
UseFixedSystemDiagnosticsAlarmIds	boolean	Das Attribut gibt an, ob Systemdiagnose-Meldungen immer die gleiche Melde-ID haben sollen (die ihrer ALID entspricht) oder ob es keine Rolle spielt, welche IDs für diese Meldungen bei ihrer Erstellung vergeben werden. Dieses Attribut ist für jede PLC des Typs S7-1500 unabhängig von der FW-Version verfügbar. Der Standardwert für neue TIA-Projekte lautet TRUE, jedoch FALSE für aktualisierte Projekte. Bei einem Wert TRUE sind die Client-Melde-IDs der Systemdiagnose in einem reservierten Bereich von maximal 512 fest definiert. Der Wechsel von FALSE nach TRUE in einem TIA-Projekt, das bereits in eine PLC geladen wurde, kann zu Folgendem führen: • Das PLC-Programm muss neu übersetzt werden • Die Online-PLC muss gestoppt werden, um zu verhindern, dass sie die geänderte Hardware-Konfiguration der PLC lädt • HMIs ohne aktivierte automatische Aktualisierung müssen erneut geladen werden Im R/H-System wird die Einstellung der Attribute zwischen den beiden PLCs synchronisiert
SuppressDeactivatingSystemDiagnosticsAlarms	boolean	Dieses Attribut gibt an, ob Systemdiagnosemeldungen über die Gerätedeaktivierung unterdrückt oder nicht unterdrückt werden sollen. Dieses Attribut ist für jede PLC des Typs S7-1500 ab FW-Version 3.0 verfügbar und ist somit für niedrigere FW-Versionen nicht sichtbar / nicht verfügbar. Der Standardwert lautet FALSE. Bei einem Wert TRUE werden Meldungen über die Gerätedeaktivierung unterdrückt. • Einträge in den Diagnosepuffer werden in keinem Fall unterdrückt. Sie können daher den Zustand aller Geräte mit dem DOE (Diagnostic and Online Editor) ermitteln. • Im R/H-System wird die Einstellung der Attribute zwischen den beiden PLCs synchronisiert

Programmcode

```

Siemens.Engineering.HW.DeviceItem plcItem = ...;
// get value of "UseFixedSystemDiagnosticsAlarmIds" via openness interface
bool isUseFixedSystemDiagnosticsAlarmIds =
(bool)plcItem.GetAttribute("UseFixedSystemDiagnosticsAlarmIds");
// get value of "SuppressDeactivatingSystemDiagnosticsAlarms" via openness interface
bool isSuppressDeactivatingSystemDiagnosticsAlarms =
(bool)plcItem.GetAttribute("SuppressDeactivatingSystemDiagnosticsAlarms");
// set value of "UseFixedSystemDiagnosticsAlarmIds" via openness interface
plcItem.SetAttribute("UseFixedSystemDiagnosticsAlarmIds", false);
// set value of "SuppressDeactivatingSystemDiagnosticsAlarms" via openness interface
plcItem.SetAttribute("SuppressDeactivatingSystemDiagnosticsAlarms", true);

```


5.11.1.2 PC-System laden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die PC-Station Plus und die SW-CPU unabhängig voneinander laden. Um die Konfiguration der PC-Station Plus zu laden, müssen Sie das Geräteelement stationmanager des Baugruppenträgers des Geräts abrufen und den download provider als Dienst abrufen. Die SW-CPU selbst muss für den gleichen Vorgang als Geräteelement abgerufen werden.

Das Laden der SW-CPU ähnelt dem Laden eines regulären PC. Weitere Informationen zum Laden einer SW-CPU finden Sie unter Herunterladen zu den PLCs (Seite 400).

Einschränkung

Das Laden von F-aktivierten PLCs wird derzeit bei einer SW-CPU genau wie bei einer HW-CPU nicht unterstützt.

Programmcode: Software- und Hardware-Komponenten laden

Sie können Software- und Hardware-Komponenten über den Dienst DownloadProvider, der über ein angegebenes DeviceItem angefordert werden kann, auf ein Gerät laden. Wenn ein DeviceItem ein ladbares Ziel darstellt, wird bei Aufruf von GetService call eine Instanz von DownloadProvider zurückgegeben. Andernfalls wird null zurückgegeben.

```
DeviceItem stationManager = dev.DeviceItems.First(p => p.PositionNumber ==
0).DeviceItems.First(a => a.PositionNumber == 125);
DownloadProvider downloadProviderStationManager =
stationManager.GetService<DownloadProvider>();
if (downloadProviderStationManager == null)
{
    //no download is possible for PC-Station
}
DeviceItem swCpu = dev.DeviceItems.First(p => p.Name == "Software PLC_1");
DownloadProvider downloadProviderSwCpu = swCpu.GetService<DownloadProvider>();
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie folgenden Programmcode, um Netzwerkparameter zu konfigurieren:

```
ConnectionConfiguration connConfig = downloadProviderStationManager.Configuration;
ConfigurationMode configurationMode = connConfig.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = configurationMode.PcInterfaces.Find("ASIX AX88179
USB 3.0 to Gigabit Ethernet Adapter", 1);
ConfigurationTargetInterface targetInterface = pcInterface.TargetInterfaces.Find("2 X2");
IConfiguration targetConfiguration = pcInterface.TargetInterfaces[0];
bool isConfigured = connConfig.ApplyConfiguration(targetInterface);
if (isConfigured)
...
```

Es ist zu beachten, dass bei einem Neustart des Zielsystems nach dem Laden der Konfiguration von PC-Station Plus das Standardverhalten von Openness **Warten** ist. Das bedeutet, dass der Openness-Codefluss gestoppt wird, bis das Laden der ersten Stufe erfolgreich ist (und das Zielsystem neu gestartet ist) oder die Zeit überschritten oder ein Fehler aufgetreten ist.

Falls Sie nicht auf den Neustart warten wollen, können Sie eine Post-Ladeoption wählen, die sich **WaitOnReboot** nennt.

```
//Post Download Configuration Delegate
DownloadConfigurationDelegate postDownloadForPcStation = downloadConfiguration =>
{
    WaitOnReboot waitOnReboot = downloadConfiguration as WaitOnReboot;
    if (waitOnReboot != null)
    {
        //In case user does not want to wait...
        waitOnReboot.CurrentSelection = WaitOnRebootSelections.NoAction;
        //In case user wants to wait... This is the default option anyway...
        //waitOnReboot.CurrentSelection = WaitOnRebootSelections.Wait;
        return;
    }
};
```

Sie müssen die SW-CPU weiterhin separat laden, aber es muss sichergestellt werden, dass zuvor die PC-Station geladen wurde.

```
DownloadResult downloadResult = null;
try
{
//WE FIRST DOWNLOAD PC-STATION
downloadResult = downloadProviderStationManager.Download(targetConfiguration,
preDownloadForPcStation, postDownloadForPcStation, DownloadOptions.Hardware);
if (DownloadResultState.Error != downloadResult.State)
{
Console.WriteLine("The download is successful for pc-station");
}
}
catch (EngineeringTargetInvocationException e)
{
Console.WriteLine("Exception Thrown, Message: " + e.Message.ToString());
}
downloadResult = null;
try
{
downloadResult = downloadProviderSwCpu.Download(targetConfiguration, preDownloadForSwCpu,
postDownloadForWinac, DownloadOptions.Hardware | DownloadOptions.Software);
if (DownloadResultState.Error != downloadResult.State)
{
Console.WriteLine("The download is successful for SW-CPU");
}
}
catch (EngineeringTargetInvocationException e)
{
Console.WriteLine("Exception Thrown, Message: " + e.Message.ToString());
}
}
```

Das Laden der SW-CPU ähnelt dem Laden eines regulären PC. Weitere Informationen zum Laden einer SW-CPU finden Sie unter Herunterladen zu den PLCs (Seite 400).

Hinweis

Das Laden von F-aktivierten PLCs wird derzeit bei einer SW-CPU genau wie bei einer HW-CPU nicht unterstützt.

Eine Lademethode stößt auch die Übersetzung an. Es ist auch möglich, ein Geräteelement unabhängig zu übersetzen, und zwar mit dem folgenden Code-Snippet:

```
ICompilable compileServiceSwCpu = swCpu.GetService<ICompilable>();
ICompilable compileServiceStationManager = stationManager.GetService<ICompilable>();
CompilerResult compileResultStationManager = compileServiceStationManager.Compile();
CompilerResult compileResultSwCpu = compileServiceStationManager.Compile();
bool compileCheck = !compileResultStationManager.State.Equals(CompilerResultState.Error);
if (compileCheck != true)
{
...
}
```

5.11.1.3 Herunterladen zu den PLCs

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet. Siehe Projekt öffnen (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie Software- und Hardwarekomponenten über DownloadProvider (Zugriff über das DeviceItem) in das PLC-Gerät laden. Wenn ein DeviceItem ein ladbares Ziel darstellt, wird bei Aufruf von GetService eine Instanz von DownloadProvider zurückgegeben. Andernfalls gibt der Dienst null zurück.

Programmcode: DownloadProvider-Dienst aus einem Geräteelement abrufen

```
DeviceItem deviceItem = ...;
DownloadProvider downloadProvider = deviceItem.GetService<DownloadProvider>();
if (downloadProvider != null)
{
    ...
}
```

Um Daten in ein Gerät zu laden, müssen Sie die Download-Methode von DownloadProvider aufrufen.

Die Parameter sind ein IConfiguration-Objekt, zwei Delegates und DownloadOptions (Hardware, Software oder Hardware und Software).

```
...
IConfiguration targetConfiguration = ...
DownloadConfigurationDelegate preDownloadDelegate = ...
DownloadConfigurationDelegate postDownloadDelegate = ...
DownloadOptions loadOption = DownloadOptions.Hardware | DownloadOptions.Software;
...
DownloadResult result = downloadProvider.Download(targetConfiguration,
preDownloadDelegate, postDownloadDelegate, loadOption);
```

Wenn sich die IP-Adresse der laufenden PLC von der offline konfigurierten IP-Adresse unterscheidet, kann ein optionaler Parameter für das Laden in eine IP-Adresse verwendet werden.

```
...
ConfigurationAddress onlineAddress = ...
DownloadResult result = downloadProvider.Download(targetConfiguration, onlineAddress,
preDownloadDelegate, postDownloadDelegate, loadOption);
```

Parameter der Download-Methode

Um Daten in ein PLC-Gerät zu laden, müssen Sie die Methode Download von DownloadProvider aufrufen. Die Download-Methode hat vier Parameter, bei denen es sich um das Objekt IConfiguration, zwei Delegates und DownloadOptions (Hardware, Software oder Hardware und Software) handelt.

Parametername	Typ	Beschreibung
configuration	Siemens.Engineering.Connection.IConfiguration	Konfiguration der Verbindung mit einem Gerät.
onlineAddress	Siemens.Engineering.Connection.ConfigurationAddress	Optionaler Parameter für das Laden in eine geänderte IP-Adresse
preDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Aufzurufender Delegate zum Prüfen der Konfiguration vor dem Ladevorgang.
postDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Aufzurufender Delegate zum Prüfen der Konfiguration nach dem Ladevorgang.
downloadOptions	Siemens.Engineering.Download.DownloadOptions	Ladeoptionen.

- Der Openness-Ladevorgang wird nur dann unterstützt, wenn die in der vorstehenden Tabelle aufgeführten Konfigurationen ordnungsgemäß von Ihnen behandelt wurden. Wenn die Konfiguration ungültig ist, wird EngineeringTargetInvocationException ausgelöst und der Ladevorgang abgebrochen.
- Da die Übersetzung ein Teil des Ladevorgangs ist, empfiehlt es sich, die Übersetzung vor dem Ladevorgang durchzuführen, um eine vollständige Rückmeldung zu den Übersetzungsergebnissen zu erhalten.

IConfiguration

Sie müssen das IConfiguration-Objekt für die Download-Methode bereitstellen. Es dient dazu, eine Verbindung zu dem angegebenen PLC-Gerät herzustellen. Die Schnittstelle IConfiguration wird von ConfigurationAddress und ConfigurationTargetInterface implementiert. Auf beide Objekte kann über die Instanz ConnectionConfiguration zugegriffen werden. Die Instanz ConnectionConfiguration kann über die Eigenschaft DownloadProvider.Connection: ConnectionConfiguration oder optional über OnlineProvider.Connection: ConnectionConfiguration ermittelt werden.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Die Konfigurierung des Objekts `ConnectionConfiguration` wird im Abschnitt `Auf Parameter einer Online-Verbindung zugreifen` (Seite 451) beschrieben.

```
...
DownloadProvider downloadProvider = null;
ConnectionConfiguration configuration = downloadProvider.Configuration;
ConfigurationMode configurationMode = configuration.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = configurationMode.PcInterfaces.Find("Intel (R)
Ethernet Connection I217-LM", 1);
IConfiguration targetConfiguration = pcInterface.TargetInterfaces[0];
...
```

ConfigurationAddress

Wenn sich die konfigurierte IP-Adresse von der bereits heruntergeladenen Online-IP-Adresse unterscheidet, muss ein zusätzlicher Parameter verwendet werden. Damit wird die richtige Online-PLC identifiziert.

```
...
DownloadProvider downloadProvider = ...
ConnectionConfiguration configuration = downloadProvider.Configuration;
ConfigurationMode configurationMode = configuration.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = configurationMode.PcInterfaces.Find("Intel (R)
Ethernet Connection I217-LM", 1);
IConfiguration targetConfiguration = pcInterface.TargetInterfaces[0];
// Create the address object to reach the PLC online.
ConfigurationTargetInterface targetConfigurationIF = targetConfiguration as
ConfigurationTargetInterface;
ConfigurationAddressComposition downloadAddresses = targetConfigurationIF.Addresses;
ConfigurationAddress onlineAddress = downloadAddresses.Create("10.119.57.223");
DownloadResult result = downloadProvider.Download(targetConfigurationIF, onlineAddress,
preDownloadDelegate, postDownloadDelegate, loadOption);
...
```

DownloadConfigurationDelegate

Sie müssen zwei Implementierungen leerer DownloadConfigurationDelegate (DownloadConfiguration downloadConfiguration) bereitstellen. Der erste Delegate wird für Konfigurationen vor dem Ladevorgang aufgerufen, der zweite nach Abschluss des Ladevorgangs. Die Delegates werden für jede Konfiguration aufgerufen, bei der eine Aktion durch den Anwender erforderlich ist. Weitere Informationen zur Handhabung von Rückrufen finden Sie unter Unterstützung von Callbacks (Seite 422). Wenn Konfigurationen nicht ordnungsgemäß verarbeitet werden, wird EngineeringTargetInvocationException ausgelöst und der Ladevorgang abgebrochen. Bestimmte Konfigurationen enthalten nur eine Information dazu, weshalb keine Aktion Ihrerseits erforderlich ist. Auch können Sie diejenigen Konfigurationen überspringen, die den Ladevorgang nicht verhindern.

Hinweis

Sie sollten Änderungen jeder Art vermeiden. Hiervon ausgenommen sind Änderungen, die für das Auflösen der Download-Konfigurationen in deren Delegate-Implementierung erforderlich sind.

Die möglichen Typen von Ladekonfigurationen werden nachstehend aufgeführt.

Konfigurationsname	Beschreibung und Eigenschaften
DownloadConfiguration	- Basisklasse für alle Konfigurationen. Sie enthält nur dann Informationen, wenn diese keiner bestimmten Konfiguration zugeordnet werden können. IConfiguration hat eine einzige Eigenschaft: - DownloadConfiguration.Message : string (schreibgeschützte Eigenschaft, die die Konfigurationsmeldung enthält)
DownloadSelectionConfiguration	- Basisklasse für alle auswählbaren Konfigurationen. Eigenschaften von DownloadSelectionConfiguration - Enthält keine weiteren Eigenschaften. In allen von dieser Klasse abgeleiteten untergeordneten Klassen muss eine Auswahl bereitgestellt werden.
DownloadCheckConfiguration	- Basisklasse für alle Konfigurationen, die geprüft oder nicht geprüft sein können. - Enthält die einzige Eigenschaft DownloadCheckConfiguration.Checked: bool<string> (Lese-/Schreib-Eigenschaft, die angibt, ob die Konfiguration geprüft oder ungeprüft ist)
DownloadPasswordConfiguration	- Basisklasse für alle Konfigurationen, die ein Passwort benötigen, um mit dem Ladevorgang fortzufahren (PLC-Schutzpasswörter und bausteingebundene PLC-Passwörter). Methoden von DownloadPasswordConfiguration: - Enthält eine einzige Methode zum Festlegen des Passworts. DownloadPasswordConfiguration.SetPassword (password: SecureString) : void - legt Passwort fest. SecureString

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Der Datentyp der Konfigurationen wird nachstehend aufgeführt.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Konfiguration	Datentyp	Beschreibung und Aktion
DownloadSelection-Configuration	StartModules	Set CurrentSelection:StartModulesSelections. Verfügbare Enumerationswerte sind NoAction (No action) StartModule (Stop module) Diese Module werden zum Laden in ein Gerät gestoppt.
	StopModules	Set CurrentSelection:StopModulesSelections. Verfügbare Enumerationswerte: NoAction (No action) StopAll (Stop all) Diese Module werden nach dem Ladevorgang gestartet.
	AllBlocksDownload	Set CurrentSelection:AllBlocksDownloadSelections. Verfügbare Enumerationswert ist DownloadAllBlocks (Download all blocks to the device) Lädt Software in das Gerät
	OverwriteSystemData	Set CurrentSelection:OverwriteSystemDataSelections. Verfügbare Enumerationswerte sind NoAction (No action) Overwrite (Download to device) Löscht und ersetzt die vorhandenen Systemdaten im Zielspeicherort.
	ConsistentBlocksDownload	Set CurrentSelection:ConsistentBlocksDownloadSelections. Verfügbare Enumerationswert ist ConsistentDownload (Consistent download) Lädt die Software in das Gerät.
	AlarmTextLibrariesDownload	Set CurrentSelection:AlarmTextLibrariesDownloadSelections. Verfügbare Enumerationswerte sind ConsistentDownload (Consistent download) NoAction (No action) Lädt alle Alarmtexte und Textlistentexte.
	ProtectionLevelChanged	Set CurrentSelection:ProtectionLevelChangedSelections. Verfügbare Enumerationswerte sind NoChange (No change) ContinueDownloading (Continue downloading to the device) Der CPU-Schutz wird auf die nächstniedrigere Stufe gesetzt.
	ActiveTestCanBeAborted	Set CurrentSelection:ActiveTestCanBeAbortedSelections. Verfügbare Enumerationswerte sind NoAction (No action) AcceptAll (Accept all) Aktive Test- und Inbetriebnahmefunktionen werden während des Ladebetriebs des Geräts abgebrochen.
	ResetModule	Set CurrentSelection:ResetModuleSelections. Verfügbare Enumerationswerte sind

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Konfiguration	Datentyp	Beschreibung und Aktion
		NoAction (No action) DeleteAll (Delete all) Setzt das Modul zurück.
	LoadIdentificationData	Set CurrentSelection:LoadIdentificationDataSelections. Verfügbare Enumerationswerte sind LoadNothing (Load nothing) LoadData (Load data) Identifikationsdaten in die PROFINET IO-Geräte und ihre Module laden.
	DifferentTargetConfiguration	Set CurrentSelection:DifferentTargetConfigurationSelections. Verfügbare Enumerationswerte sind NoAction (No action) AcceptAll (Accept all) Gibt den Unterschied zwischen konfiguriertem Modul und Zielmodul (online) an
	InitializeMemory	Set CurrentSelection:InitializeMemorySelections. Verfügbare Enumerationswerte: NoAction (No action) AcceptAll (Accept all) Dieser Datentyp dient zum Initialisieren des Speichers.
	ExpandDownload	Set CurrentSelection: ExpandDownloadSelections. Verfügbare Enumerationswerte: NoAction (No action) Download (Download)}} Das Laden muss über Ihre Auswahl hinaus erweitert werden.
	ActiveTestCanPreventDownload	Set/Get CurrentSelection:ActiveTestCanPreventDownloadSelections Verfügbare Enumerationswerte: NoAction (No action) AcceptAll (Accept all) Aktive Test- und Inbetriebnahmefunktionen können den Ladevorgang verhindern.
	DataBlockReinitialization	Set CurrentSelection:DataBlockReinitializationSelections. Verfügbare Enumerationswerte: StopPlcAndReInitialize (Alle Datenwerte, einschließlich der Retain-Daten, werden beim Laden mit ihren definierten Startwerten initialisiert. Stellen Sie die PLC vor dem Laden auf STOP). NoAction (Keine Aktion bei "Herunterladen mit Neuinitialisierung"). Unterschiedliche Speicherreserven im Online- und Offline-Block verhindern den "Download ohne Neuinitialisierung".

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Konfiguration	Datentyp	Beschreibung und Aktion
DownloadCheckConfiguration	CheckBeforeDownload	Eigenschaft Set <code>IsChecked:bool</code> Prüft vor dem Laden in das Gerät.
	UpgradeTargetDevice	Eigenschaft Set <code>IsChecked:bool</code> Prüft die unterschiedlichen Projektversionen im konfigurierten Gerät und im Zielgerät (online).
	OverWriteHMIData	Eigenschaft Set <code>IsChecked:bool</code> Überschreibt, wenn Objekt online vorhanden ist.
	FitHMIComponents	Eigenschaft Set <code>IsChecked:bool</code> Komponenten mit einer anderen Version werden auf dem Zielgerät installiert.
	TurnOffSequence	Eigenschaft Set <code>IsChecked:bool</code> Schaltet die Sequenz vor dem Laden aus.
	OverwriteTargetLanguage	Eigenschaft Set <code>IsChecked:bool</code> Zum Unterscheiden der Einstellungen für Projekt und PLC-Programmierung
	DowngradeTargetDevice	Eigenschaft Set <code>IsChecked:bool</code> Zum Erwärmen der verschiedenen Datenformate in Online- und Offline-Projekten.
DownloadPassword-Configuration	ModuleReadAccessPassword	Methode Set <code>password</code> via <code>SetPassword(password:SecureString)</code> . Passwort eingeben, um Lesezugriff auf das Modul zu erhalten.
	ModuleWriteAccessPassword	Methode Set <code>password</code> via <code>SetPassword(password:SecureString)</code> . Passwort eingeben, um Schreibzugriff auf das Modul zu erhalten.
	BlockBindingPassword	Methode Set <code>password</code> via <code>SetPassword(password:SecureString)</code> . Methode für die Konfiguration eines bausteingebundenen Passworts.
	PlcMasterSecretPassword	Methode Set <code>password</code> via <code>SetPassword(password:SecureString)</code> . Für die Eingabe des Master-Secret-Passworts der PLC

Hinweis

Die Konfigurationen, die die Eingabe eines Passworts ermöglichen, haben ein zusätzliches Attribut

ACHTUNG

Beachten Sie bitte, dass die Ladekonfigurationen den Konfigurationen in den Dialogen mit der Ladevorschau und den Ladeergebnissen ähneln, wenn Sie mit der Benutzeroberfläche des TIA Portals arbeiten.

**WARNUNG**

Der API-Benutzer ist verantwortlich für die Sicherheitsmaßnahmen zur Handhabung von Passwörtern durch Code.

IsSecureCommunication

Das Attribut liefert "TRUE" im Falle einer "Sicheren Kommunikations"-Verbindung (TLS-Handshake) zur PLC. In allen anderen Fällen liefert das Attribut "FALSE". Weitere Informationen über IsSecureCommunication siehe Unterstützung der sicheren S7-Kommunikation mit TLS (Seite 492)

Dies betrifft also:

- ModuleReadAccessPassword
- ModuleWriteAccessPassword
- BlockBindingPassword
- PlcMasterSecretPassword

```
bool isSecureCommunication = moduleReadAccessPassword.IsSecureCommunication;
if (isSecureCommunication == true)
{
    // Secure communication with (Tls handshake)
    ...
}
```

Nicht behandelte Ausnahmen in den Delegates

Nicht verarbeitete Konfiguration, die den Ladevorgang verhindern kann, verursacht eine `EngineeringTargetInvocationException` und bricht den Ladevorgang ab. Bei einer nicht verarbeiteten Ausnahme im Delegate wird eine `EngineeringDelegateInvocationException` ausgelöst.

Für die Implementierung von `PreDownloadDelegate` können Sie den nachstehenden Beispielcode verwenden:

```
private static void PreConfigureDownload(DownloadConfiguration downloadConfiguration)
{
    StopModules stopModules = downloadConfiguration as StopModules;
    if (stopModules != null)
    {
        stopModules.CurrentSelection = StopModulesSelections.StopAll; // This selection will set
        PLC into "Stop" mode
        return;
    }
    AlarmTextLibrariesDownload alarmTextLibraries = downloadConfiguration as
    AlarmTextLibrariesDownload;
    if (alarmTextLibraries != null)
    {
        alarmTextLibraries.CurrentSelection =
        AlarmTextLibrariesDownloadSelections.ConsistentDownload;
        return;
    }
    BlockBindingPassword blockBindingPassword = downloadConfiguration as BlockBindingPassword;
    if (blockBindingPassword != null)
    {
        SecureString password = ...; // Get Binding password from a secure location
        blockBindingPassword.SetPassword(password);
        return;
    }
    CheckBeforeDownload checkBeforeDownload = downloadConfiguration as CheckBeforeDownload;
    if (checkBeforeDownload != null)
    {
        checkBeforeDownload.Checked = true;
        return;
    }
    ConsistentBlocksDownload consistentBlocksDownload = downloadConfiguration as
    ConsistentBlocksDownload;
    if (consistentBlocksDownload != null)
    {
        consistentBlocksDownload.CurrentSelection =
        ConsistentBlocksDownloadSelections.ConsistentDownload;
        return;
    }
    ModuleWriteAccessPassword moduleWriteAccessPassword = downloadConfiguration as
    ModuleWriteAccessPassword;
    if (moduleWriteAccessPassword != null)
    {
        SecureString password = ...; // Get PLC protection level password from a secure location
        moduleWriteAccessPassword.SetPassword(password);
        return;
    }
    throw new NotSupportedException(); // Exception thrown in the delegate will cancel download
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Für die Implementierung von `PostDownloadDelegate` können Sie den nachfolgenden Beispielcode verwenden:

```
private static void PostConfigureDownload(DownloadConfiguration downloadConfiguration)
{
    StartModules startModules = downloadConfiguration as StartModules;
    if (startModules != null)
    {
        startModules.CurrentSelection = StartModulesSelections.StartModule; // Sets PLC in "Run"
        mode
    }
}
```

DownloadOptions

Sie müssen die Ladeoptionen über `DownloadOptions`, durch enum gekennzeichnet, angeben. Dieser Parameter bestimmt den Typ des durchzuführenden Ladevorgangs, z.B. Hardware, Software, `SoftwareOnlyChanges` oder eine Kombination davon.

```
[Flags]
public enum DownloadOptions
{
    None = 0, // Download nothing
    Hardware = 1, // Download hardware. Complete download when used alone, delta download when
    used in combination with Software or SoftwareOnlyChanges.
    Software = 2, // Download software all. Do not combine with SoftwareOnlyChanges
    SoftwareOnlyChanges = 4 // Download software only changes. Do not combine with Software
}
```

Mögliche <code>DownloadOptions</code> -Kombinationen (C#-Code)	Systemreaktionen
keine	Ausnahme
Hardware	HW vollständig
Software	SW vollständig
Hardware Software	HW Delta, SW vollständig
SoftwareOnlyChanges	SW Delta
Hardware SoftwareOnlyChanges	HW Delta, SW Delta
Software SoftwareOnlyChanges	Ausnahme
Hardware Software SoftwareOnlyChanges	Ausnahme
(<code>DownloadOptions</code>)11	Ausnahme

DownloadResult

Das von der Aktion `Download` zurückgegebene `DownloadResult` liefert Rückmeldung über den Zustand der geladenen Objekte.

Für den Aufruf des Ladevorgangs können Sie den nachfolgenden Beispielcode verwenden:

```
[STAThread]
static void Main()
{
    ...
    DownloadProvider downloadProvider = ...;
    IConfiguration targetConfiguration = ...;
    DownloadConfigurationDelegate preDownloadDelegate = PreConfigureDownload;
    DownloadConfigurationDelegate postDownloadDelegate = PostConfigureDownload;
    DownloadResult result = downloadProvider.Download(targetConfiguration,
    preDownloadDelegate, postDownloadDelegate, DownloadOptions.Hardware |
    DownloadOptions.Software);
    if (result.State == DownloadResultState.Error)
    {
        // Handle error state
    }
    WriteDownloadResults(result);
    ...
}
private static void PreConfigureDownload(DownloadConfiguration downloadConfiguration)
{
    ...
}
private static void PostConfigureDownload(DownloadConfiguration downloadConfiguration)
{
    ...
}
private void WriteDownloadResults(DownloadResult result)
{
    Console.WriteLine("State:" + result.State);
    Console.WriteLine("Warning Count:" + result.WarningCount);
    Console.WriteLine("Error Count:" + result.ErrorCount);
    RecursivelyWriteMessages(result.Messages);
}
private void RecursivelyWriteMessages(DownloadResultMessageComposition messages, string
indent = "")
{
    indent += "\t";
    foreach (DownloadResultMessage message in messages)
    {
        Console.WriteLine(indent + "DateTime: " + message.DateTime);
        Console.WriteLine(indent + "State: " + message.State);
        Console.WriteLine(indent + "Message: " + message.Message);
        RecursivelyWriteMessages(message.Messages, indent);
    }
}
```

5.11.1.4 Herunterladen der PLC in einen Windows-Ordner

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet. Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können TIA Portal Openness nutzen, um eine PLC in einen Windows-Ordner im Windows-Dateisystem herunterzuladen. Es ist möglich, den Prozess der Erstellung der SIMATIC-Speicherkarte zusammen mit dem PLC-Programm zu automatisieren, ohne dass eine Verbindung zur PLC oder zu PLCSIM Advanced erforderlich ist.

Die Funktion gilt für folgende Produkte/Geräte:

- S7-1500 CPU einschließlich Safety (FW $\geq$ 2.1)
- S7-1500R/H CPU einschließlich Safety (HF)
- ET 200SP CPU einschließlich Safety
- ET 200pro CPU einschließlich Safety
- S7-1500 Software Controller einschließlich Safety
- ET 200SP Open Controller einschließlich Safety
- S7-1200 CPU einschließlich Safety (FW $\geq$ V4.2)

Parameter

DownloadProvider.Download() und RHDDownloadProvider.Download() akzeptiert die folgende Definition von Parametern. Beide Methoden geben DownloadResult zurück.

Parametername	Typ	Beschreibung
directoryInfo	System.IO.DirectoryInfo	Gibt den Pfad des Windows-Ordners zum Speichern der Konfiguration an
preDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Gibt den Delegate an, der aufgerufen wird, um die Konfiguration vor dem Laden zu prüfen.

Weitere Informationen zu DownloadConfigurationDelegate und DownloadResult finden Sie unter Herunterladen zu den PLCs (Seite 400).

Die folgenden Konfigurationen werden zum Laden der PLC in den Windows-Ordner unterstützt:

Datentyp	Aktion
OverwriteOnMemoryCard	Einstellung von CurrentSelection: OverwriteOnMemoryCardSelections Verfügbare Enum-Werte: - NoAction (Keine Aktion) - Load (In Gerät herunterladen)
TargetForSoftware	Einstellung von CurrentSelection: TargetForSoftwareSelections Verfügbare Enum-Werte: - CPU (CPU) - PlcSimulationAdvanced (Erweiterte PLC-Simulation)
ConsistentBlocksDownload	Einstellung von CurrentSelection: ConsistentBlocksDownloadSelections Verfügbare Enum-Werte: ConsistentDownload (Konsistentes Laden)
AlarmTextLibrariesDownload	Einstellung von CurrentSelection: AlarmTextLibrariesDownloadSelections Verfügbare Enum-Werte: - ConsistentDownload (Konsistentes Laden) - NoAction (Keine Aktion)
SafetyProgram	Einstellung von CurrentSelection: SafetyProgramSelections Verfügbare Enum-Werte: ConsistentDownload (Konsistentes Laden)

Weitere Informationen zu unterstützten Typen für die Konfiguration von Downloads finden Sie unter Herunterladen zu den PLCs (Seite 400)

Programmcode: Spezifikationen für das Laden in den Windows-Ordner

Ändern Sie den folgenden Programmcode, um auf DownloadProvider für Standard-PLC zuzugreifen:

```
DeviceItem deviceItem = ...;
DownloadProvider downloadProvider = deviceItem.GetService<DownloadProvider>();
downloadProvider.Download(new DirectoryInfo(@"c:\tmp\card1"),
DownloadConfigurationDelegate);
```

Hinweis

Sie müssen den lokalen Windows-Ordner im Windows-Dateisystem und eine Delegierungsmethode angeben.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie den folgenden Programmcode, um auf RHDownloadProvider für das R/H-System zuzugreifen:

```
Device device = ...;
RHDownloadProvider rhDownloadProvider = device.GetService<RHDownloadProvider>();
rhDownloadProvider.Download(new DirectoryInfo(@"c:\tmp\card1"),
DownloadConfigurationDelegate);
```

Hinweis

Obwohl R/H-Systeme aus zwei PLCs bestehen, ist nur eine einzige Download-Aktion erforderlich. Die gespeicherte Konfiguration wird verwendet, um die primäre und die Backup-PLC gleichzeitig zu aktualisieren.

Damit der Download erfolgreich durchgeführt werden kann, muss das Openness-Programm die folgenden Konfigurationen verwalten

```
private static void DownloadConfigurationDelegate(DownloadConfiguration
downloadConfiguration)
{
    if (downloadConfiguration is TargetForSoftware configTargetForSoftware)
    {
        configTargetForSoftware.CurrentSelection =TargetForSoftwareSelections.CPU;
    }
    else if (downloadConfiguration is OverwriteOnMemoryCard configOverwriteOnMemoryCard)
    {
        configOverwriteOnMemoryCard.CurrentSelection = OverwriteOnMemoryCardSelections.Load;
    }
    else if (downloadConfiguration is ConsistentBlocksDownload configConsistentBlocksDownload )
    {
        configConsistentBlocksDownload.CurrentSelection =
        ConsistentBlocksDownloadSelections.ConsistentDownload;
    }
    else if (downloadConfiguration is AlarmTextLibrariesDownload
configAlarmTextLibrariesDownload)
    {
        configAlarmTextLibrariesDownload.CurrentSelection =
        AlarmTextLibrariesDownloadSelections.ConsistentDownload;
    }
    else if (downloadConfiguration is SafetyProgram safetyProgram)
    {
        safetyProgram.CurrentSelection = SafetyProgramSelections.ConsistentDownload;
    }
}
```

Programmcode: Beispiel für DownloadResult

```
String directory = ...;
var downloadResult = downloadProvider.Download(new DirectoryInfo(directory),
DownloadConfigurationDelegate);
PrintAllMessages(downloadResult.Messages, 0, downloadResult);
```

5.11.1.5 Master Secret in eine PLC laden

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Mit TIA Portal Openness können Sie Master Secrets über PlcMasterSecretPassword mit Zugriff über die Download-Konfiguration in die PLC laden. Für das Laden müssen Sie ein leeres DownloadConfigurationDelegate (DownloadConfiguration downloadConfiguration) für die Konfiguration vor dem Download implementieren.

Dieses Delegate wird für jede Konfiguration aufgerufen, die eine Handlung des Benutzers erfordert. Die PlcMasterSecretPassword-Konfiguration verarbeitet die Benutzerhandlung für die Eingabe des Master-Secret-Passworts.

Weitere Informationen zur Handhabung von Rückrufen und die Unterstützung des Downloads in PLC-Geräte finden Sie unter Unterstützung von Callbacks (Seite 422) und Herunterladen zu den PLCs (Seite 400).

Programmcode: PlcMasterSecretPassword Download

Mit dem folgenden Beispielcode können Sie Software- und Hardware-Komponenten über den Dienst DownloadProvider in ein Gerät laden. Für Informationen über IsSecureCommunication siehe Unterstützung der sicheren S7-Kommunikation mit TLS (Seite 492)

```
internal bool DownloadDevice(DeviceItem deviceItem)
{
    try
    {
        //Download software and hardware components to a device via DownloadProvider service that
        //can be acquired from a given DeviceItem. If a DeviceItem represents a download able
        //target an instance of DownloadProvider will be returned on GetService call. Otherwise
        //it will be null.
        DownloadProvider myDownloadProvider = deviceItem.GetService<DownloadProvider>();
        if (myDownloadProvider == null)
        {
            return false;
        }
        //Preparation for IConfigurable object to establish a connection to the device.
        IConfiguration
        //interface is implemented by ConfigurationAddress and ConfigurationTargetInterface. Both
        //objects can be accessed via ConnectionConfiguration instance. ConnectionConfiguration
        //instance can be acquired from DownloadProvider.Configuration
        ConnectionConfiguration connConfiguration = myDownloadProvider.Configuration;
        ConfigurationMode confMode = connConfiguration.Modes.Find("PN/IE");
        ConfigurationPcInterface pcInterface = ("Intel(R) Ethernet Connection I217-LM", 1);
        ConsoleOutput.PrintInterfaces(connConfiguration);
        //In order to execute a download to a device, Download method of DownloadProvider have to
        //be called. The parameters are an IConfiguration object, two delegates and DownloadOptions
        //(Hardware, Software or Hardware and Software).
        IConfiguration targetConfiguration = pcInterface.TargetInterfaces[0];
        DownloadConfigurationDelegate preDownloadConfigurationDelegate = PreConfigureDownload;
        DownloadConfigurationDelegate postDownloadConfigurationDelegate = PostConfigureDownload;
        DownloadOptions downloadOptions = DownloadOptions.Hardware | DownloadOptions.Software;
        // Apply the the configuration and start the download
        myDownloadProvider.Configuration.ApplyConfiguration(pcInterface.TargetInterfaces.First());
        var downLoadResult = myDownloadProvider.Download(targetConfiguration,
        preDownloadConfigurationDelegate,
        postDownloadConfigurationDelegate, downloadOptions);
        return downLoadResult != null && downLoadResult.State != DownloadResultState.Error;
    }
    catch (EngineeringTargetInvocationException ex)
    {
        //Exception catch-ed on purpose. Client can write own code to handle the exception
        //In case of PLC Master Secret exception will be thrown if:
        // 1) If a different Master Secret has already been set in the PLC.
        //Exception Message: An error has occurred during download: 'A different Master Secret
        //has already been set. The PLC has to be reset!
        //Possibly provide custom exception handling here
        Console.WriteLine("An exception during download was thrown:");
        Console.WriteLine(ex.DetailMessageData);
        return false;
    }
}
```

Zur Handhabung von `PlcMasterSecretPassword` in `PreDownloadDelegate` können Sie den folgenden Beispielcode verwenden:

```
private static void PreConfigureDownload(DownloadConfiguration downloadConfiguration)
{
    ... ..
    // Set the PLC Master Secret password configured in the offline project's PLC,
    // when the password was never set in the real PLC
    if (downloadConfiguration is PlcMasterSecretPassword plcMasterSecret)
    {
        // Get the PLC Master Secret password from a secure location
        SecureString plcMasterSecretPsw = ...;
        plcMasterSecret.SetPassword(plcMasterSecretPsw);
        return;
    }
    catch (EngineeringTargetInvocationException ex)
    {
        // Exception catch-ed on purpose. Client can write own code to handle the exception
        // Exception will be thrown if:
        // 1) plcMasterSecret password is null.
        // Exception Message: The argument "password" may not be null
        // 2) plcMasterSecret password is an empty string.
        // Exception Message: The password provided is invalid
        // 3) plcMasterSecret password is wrong.
        // Exception Message: The password provided is invalid
        // Possibly provide custom exception handling here
        Console.WriteLine("PlcMasterSecretPassword has thrown an exception:");
        Console.WriteLine(ex.DetailMessageData);
        return;
    }
    ... ..
    throw new NotSupportedException();
    // Exception thrown in the delegate will cancel download
}
```

Hinweis

Bei einer nicht verarbeiteten Ausnahme im Delegate wird eine `EngineeringDelegateInvocationException` ausgelöst.

5.11.1.6 Master Secrets in PLCs verwalten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer Anwendung TIA Portal Openness ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können TIA Portal Openness für den Schutz sensibler PLC-Konfigurationsdaten verwenden. Sie können einen PLC Master-Secret-Key konfigurieren, der für die Verschlüsselung der Zertifikatkonfiguration verwendet wird.

Außerdem können Sie diesen deaktivieren, wenn der Schutz der PLC-Konfigurationsdaten keine Priorität hat.

Sie können in TIA Portal Openness den PLC Master Secret Key auch über die Hardwarefunktion PlcMasterSecretConfigurator in der Deviceltem-CPU des Geräts konfigurieren. Diese Funktion bietet Maßnahmen für den Schutz, die Aufhebung des Schutzes und das Ändern des Master Secret Key.

- void Protect(SecureString password)
- void Unprotect(SecureString password)
- void Unprotect()
- void ChangePassword(SecureString oldPassword, SecureString newPassword)
- attribute MasterSecretConfiguration MasterSecretConfiguration{get;}
- void Reset()

```
enum MasterSecretConfiguration
{
    None = 0 , // The checkbox option "Protect the PLC Configuration data..." is unchecked.
    WithPassword = 1, // The PLC Master Secret is configured.
    WithoutPassword = 2 // The checkbox option "Protect the PLC Configuration data..." is
    //checked and the PLC Master Secret is not configured.
}
```

Master-Secret-Key einer PLC konfigurieren

Das Konfigurieren des Master Secret Key für eine PLC ist aus Sicherheitsgründen als SecureString implementiert. Es ist nur möglich, die Passwörter zu setzen / zu ändern. Das Auslesen der Passwörter wird nicht unterstützt.

```
PlcMasterSecretConfigurator plcMasterSecretConfigurator =
    plcCpu.GetService<PlcMasterSecretConfigurator>();
plcMasterSecretConfigurator.Protect(StringToSecureString("TiaPassword123"));
plcMasterSecretConfigurator.ChangePassword(StringToSecureString("TiaPassword123"),
    StringToSecureString("TiaPassword1234"));
public SecureString StringToSecureString(string value)
{
    SecureString secureStr = new SecureString();
    if (!string.IsNullOrEmpty(value))
    {
        for(int i = 0; i < value.Length; i++)
        {
            secureStr.AppendChar(value[i]);
        }
    }
    return secureStr;
}
```

Funktionalität für PLC Master Secret deaktivieren

Das Deaktivieren des PLC-Master-Secret-Passworts ist mit Hilfe überladener Unprotect-Methoden implementiert. Sie können anhand der Master-Secret-Konfiguration die PLC-Master-Secret-Funktion mit und

ohne Master Secret Key deaktivieren. Das Auslesen der Passwörter wird nicht unterstützt.

```
//Unprotect method that will be called when PlcMasterSecret is not configured
plcMasterSecretConfigurator.Unprotect();

//Unprotect method that will be called when PlcMasterSecret is configured
plcMasterSecretConfigurator.Unprotect(StringToSecureString("TiaPassword123"));
```

Funktionalität für PLC Master Secret zurücksetzen

Wenn Sie den Master Secret Key vergessen, wird die Konfiguration durch das Zurücksetzen entfernt und die Zertifikate gehen verloren.

```
plcMasterSecretConfigurator.Protect(StringToSecureString("TiaPassword123"));
plcMasterSecretConfigurator.Reset();
```

5.11.1.7 Ein PLC Master Secret in einer PLC setzen / löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness das PlcMasterSecret für eine PLC setzen oder zurücksetzen. Um die Funktionen in TIA Portal Openness auszuführen, wurde OnlineProvider um neue Methoden erweitert:

- SetPlcMasterSecret()
- ResetPlcMasterSecret()

Das SetPlcMasterSecret() unterstützt folgenden Parameter:

Parameter	Typ	Beschreibung
securePassword	SecureString	Master-Secret-Passwort.

Hinweis

Das PLC Master Secret kann nicht mit einem NULL-Passwort gelöscht oder zurückgesetzt werden.

Programmcode: PlcMasterSecret setzen und rücksetzen

Ändern Sie den folgenden Programmcode, um PlcMasterSecret über den Dienst OnlineProvider zu setzen oder zurückzusetzen:

```
DeviceItem deviceItem = ...;  
SecureString password = ...;  
OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
```

Ändern Sie den folgenden Programmcode, um PlcMasterSecret zu setzen:

```
DeviceItem deviceItem = ...;  
SecureString password = ...;  
OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();  
onlineProvider.SetPlcMasterSecret(password);
```


Ändern Sie den folgenden Programmcode, um PlcMasterSecret zurückzusetzen:

```
DeviceItem deviceItem = ...;  
OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();  
onlineProvider.ResetPlcMasterSecret();
```

Hinweis

Das Zurücksetzen des PLC Master Secrets entspricht der Löschfunktion im Dialog "Online & Diagnose".

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.1.8 PLC starten und stoppen**Voraussetzungen**

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Der PLC ist offline.

Verwendung

Bei Interaktionen mit TIA Portal über Openness API kann es notwendig sein, die Betriebsart des PLC zu ändern. TIA Portal Openness bietet die Möglichkeit, den Betriebszustand des PLC entweder in Start oder in Stop zu ändern.

Programmcode

Ändern Sie folgenden Programmcode, um den Betriebszustand des PLC auf STOP einzustellen.

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
{
    StopModules stopModules = downloadConfiguration as StopModules;
    if (stopModules != null)
    {
        // Puts PLC in "Stop" mode
        stopModules.CurrentSelection = StopModulesSelections.StopAll;
    }
}
```

Ändern Sie folgenden Programmcode, um den Betriebszustand des PLC auf START einzustellen.

```
public void ConfigurePostDownload(DownloadConfiguration downloadConfiguration)
{
    StartModules startModules = downloadConfiguration as StartModules;
    if (startModules != null)
    {
        // Puts PLC in "Start" mode
        startModules.CurrentSelection = StartModulesSelections.StartModule;
    }
}
```

5.11.1.9 Unterstützung von Callbacks

Voraussetzungen

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Bestimmte API-Methoden erfordern während ihrer Ausführung eine Interaktion mit dem benutzerdefinierten Anwendungscode. Delegates dienen zur Handhabung dieser Callback-Aktionen im benutzerdefinierten Anwendungscode. Sie müssen eine Methode mit einer kompatiblen Signatur implementieren und als Parameter delegate an die Aktion übergeben. Für die Ausführung ruft das TIA Portal die implementierten Methoden auf.

Programmcode

```
// This delegate is declared in Siemens.Engineering.dll
public delegate void
Siemens.Engineering.Download.DownloadConfigurationDelegate(Siemens.Engineering.Download.Co
nfigurations.DownloadConfiguration configuration);
...
```

Beispiel für einen benutzerdefinierten Anwendungscode mit Verwendung und Implementierung von delegate:

```
[STAThread]
static void Main()
{
    ...
    DownloadProvider downloadProvider = ...;
    IConfiguration targetConfiguration = ...;
    DownloadConfigurationDelegate preDownloadDelegate = PreConfigureDownload;
    DownloadConfigurationDelegate postDownloadDelegate = PostConfigureDownload;
    DownloadResult result = downloadProvider.Download(targetConfiguration,
preDownloadDelegate, postDownloadDelegate, DownloadOptions.Hardware |
DownloadOptions.Software);
    ...
}

//This method will be called back by TIA Portal
private static void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
{
    // Work with the parameter
}

//This method will be called back by TIA Portal
private static void ConfigurePostDownload(DownloadConfiguration downloadConfiguration)
{
    // Work with the parameter
}
```

Hinweis

Das Attribut **STAThread** stellt sicher, dass die Delegates im Haupt-Thread der Ausführung aufgerufen werden.

5.11.1.10 PLC durch Passwort schützen

Voraussetzungen

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Der PLC ist offline.

Verwendung

Bei Interaktionen mit TIA Portal über Openness API kann es notwendig sein, den Schutzgrad des PLC zu ändern. TIA Portal Openness bietet die Möglichkeit, den PLC durch ein Passwort zu schützen. Das Passwort kann sowohl für lesegeschützte als auch für schreibgeschützte PLCs eingestellt werden.

Programmcode

Ändern Sie folgenden Programmcode für lesegeschützte PLCs.

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
{
    ModuleReadAccessPassword moduleReadAccessPassword = downloadConfiguration
asModuleReadAccessPassword;
    if (moduleReadAccessPassword != null)
    {
        SecureString password = ...; // Get password from a secure location
        moduleReadAccessPassword.SetPassword(password); // enter the password to gain
full access
    }
}
```

Ändern Sie den folgenden Programmcode für schreibgeschützte PLCs:

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
{
    ModuleWriteAccessPassword moduleWriteAccessPassword = downloadConfigurationas
ModuleWriteAccessPassword;
    if (moduleWriteAccessPassword != null)
    {
        SecureString password = ...; // Get password from a secure location
        moduleWriteAccessPassword.SetPassword(password); // enter the password to gain full
access
    }
}
```

**WARNUNG**

Der API-Benutzer ist verantwortlich für die Sicherheitsmaßnahmen zur Handhabung von Passwörtern durch Code.

5.11.1.11 Handhabung bausteingebundener PLC-Passwörter

Voraussetzungen

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Der PLC ist offline.

Verwendung

TIA Portal Openness unterstützt die Datenbindung von Passwörtern für Kundenanwendungen. TIA Portal Openness bietet dem Kunden einen Weg, ein bausteingebundenes Passwort anzugeben. Zum Beispiel kann ein bausteingebundenes Passwort auf der Klasse `DownloadPasswordConfiguration` durch Aufrufen der Methode `SetPassword` konfiguriert werden.

Hinweis

Wenn Sie den Ladevorgang mit einem Passwort schützen wollen, muss bei jedem Aufruf der Download-Funktion ein Passwort angegeben werden. Das gilt unabhängig davon, ob das Gerät bereits konfiguriert ist. Nach erfolgreicher Eingabe des Passworts für eine gegebene Konfiguration werden alle nachfolgenden Aufrufe von `SetPassword` ignoriert.

Programmcode

Ändern Sie folgenden Programmcode:

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
{
    DownloadPasswordConfiguration downloadPasswordConfiguration = downloadConfiguration as
    DownloadPasswordConfiguration;
    if(downloadPasswordConfiguration != null &&
    downloadPasswordConfiguration.Message.Contains("block_1"))
    {
        SecureString password = ...; // Get password from a secured location
        downloadPasswordConfiguration.SetPassword(password);
    }
}
```

5.11.1.12 Hochladen der PLC

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie über StationUploadProvider eine Station in ein Projekt laden. Die Instanz von StationUploadProvider wird bei GetService() zurückgegeben, das mit einem project aufgerufen wird, um einen Upload auszuführen.

Die Upload-Station wird in Bezug auf die Gerätegruppe nicht unterstützt.

Programmcode: StationUploadProvider-Dienst von einem Projekt abrufen

```
Project myProject = ...;
StationUploadProvider uploadProviderForProject =
myProject.GetService<StationUploadProvider>();
if (uploadProviderForProject != null)
{
    ...
}
```

Parameter der Upload-Methode

Um das Laden in ein PLC-Gerät durchzuführen, können Sie die Upload-Methode StationUpload() von StationUploadProvider. aufrufen. Die Parameter dieser Upload-Methode sind ConfigurationAddress und UploadConfigurationDelegte .

Da mit der Methode StationUpload Software, Hardware und Dateien geladen werden, sind die UploadOptions optional.

Parametername	Typ	Beschreibung
configurationAddress	Siemens.Engineering.Connection.ConfigurationAddress	Adresse des zu ladenden Geräts
uploadConfigurationDelegate	Siemens.Engineering.Upload.UploadConfigurationDelegate	Delegate, der aufgerufen wird, um die Konfiguration vor dem Laden zu prüfen
uploadOptions	Siemens.Engineering.Upload.UploadOptions	Upload-Optionen

Parameter 1: ConfigurationAddress

Sie müssen das Objekt ConfigurationAddress für die Upload-Methode angeben. Das Adressobjekt dient dazu, eine Verbindung zum angegebenen PLC-Gerät herzustellen, in das geladen werden soll. Das ConfigurationAddress -Objekt muss in der ConnectionConfiguration des StationUploadProvider erstellt werden.

Die Configuration enthält eine Liste unterstützter Modes. Einer der Modes muss zur Verwendung beim Laden ausgewählt werden. Der ausgewählte ConfigurationMode enthält eine Liste aller lokalen PcInterfaces, die den ausgewählten Mode unterstützen. Eine dieser Schnittstellen muss ausgewählt werden. Die gewünschte Adresse kann in der Address-Sammlung der ausgewählten ConfigurationPcInterface erstellt werden.

Ändern Sie zum Erstellen eines Adressobjekts den folgenden Code:

```
...
StationUploadProvider uploadProvider = null;
...
ConnectionConfiguration configuration = uploadProvider.Configuration;
ConfigurationMode configurationMode = configuration.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = configurationMode.PcInterfaces.Find("Intel (R)
Ethernet Connection I217-LM", 1);
// "Create an address. This "ConfigurationAddress" is used as parameter for upload."
ConfigurationAddress uploadAddress = pcInterface.Addresses.Create("192.68.0.1");
...
```

Upload des Projekts

Sie können den Stations-Upload durch Aufrufen der Aktion StationUpload starten.

Die folgenden Parameter sind obligatorisch:

- ConfigurationAddress: Die Adresse des Geräts, in das geladen werden soll
- UploadConfigurationDelegate: Der Rückruf zur Behandlung von Lade-Inhibits

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

```

...
StationUploadProvider uploadProvider = null;
Device uploadedObject = null;
...
UploadConfigurationDelegate preUploadDelegate = PreConfigureUpload;
UploadResult result = uploadProvider.StationUpload(uploadAddress, preUploadDelegate);
// The uploaded device
uploadedObject = result.UploadedStation;
if (uploadedObject == null)
{
    ...
}
internal void PreConfigureUpload(UploadConfiguration uploadConfiguration)
{
    ...
}

```

Parameter2: UploadConfigurationDelegate

Sie müssen eine Implementierung von void UploadConfigurationDelegate (UploadConfiguration uploadConfiguration) bereitstellen. Der Delegate wird für Konfigurationen vor dem Laden aufgerufen. Der Delegate wird für jede Konfiguration aufgerufen, bei der eine Aktion durch den Anwender erforderlich ist. Weitere Informationen zur Handhabung von Rückrufen finden Sie unter Unterstützung von Callbacks (Seite 422). Bestimmte Konfigurationen enthalten nur eine Information, weshalb keine Aktion durch den Anwender erforderlich ist.

Die möglichen Typen von Ladekonfigurationen werden nachstehend aufgeführt:

Konfigurationsname	Beschreibung und Eigenschaften
UploadConfiguration	- Basisklasse für alle Konfigurationen. Sie enthält Informationen im Message-Attribut - Enthält die einzige Eigenschaft UploadConfiguration.Message : string (schreibgeschützte Eigenschaft, die die Konfigurationsmeldung enthält)
UploadPasswordConfiguration	- Abgeleitet von UploadConfiguration - Basisklasse für alle Konfigurationen, die ein Passwort für den Ladevorgang benötigen. - Enthält eine einzige Methode zum Festlegen des Passworts. UploadPasswordConfiguration.SetPassword (password: SecureString) : void - Passwort festlegen
UploadSelectionConfiguration	- Abgeleitete Klasse von UploadConfiguration - Enthält keine weiteren Eigenschaften.

Der Datentyp der Konfigurationen wird nachstehend aufgeführt.

Konfiguration	Datentyp	Beschreibung und Aktion
UploadPasswordConfiguration	ModuleReadAccessPassword	Methode <code>Set password</code> via <code>SetPassword(password:SecureString)</code> . Passwort eingeben, um Lesezugriff auf das Modul zu erhalten.
	PasswordReadAccess	Methode <code>Set password</code> via <code>SetPassword(password:SecureString)</code> . Passwort für den Softwareupload in klassischen PLCs eingeben, um Lesezugriff auf das Modul zu erhalten.
UploadSelectionConfiguration	UploadMissingProducts	<code>Set CurrentSelection:UploadMissingProductsSelections</code> Verfügbare Enumerationswerte: <code>TryUpload (Consistent upload)</code> <code>NoAction (No action)</code> Eine Auswahl für den Upload angeben.

Unterstützung eines fehlersicheren Passworts ist nicht notwendig. Für den Lesezugriff beim Laden in einen F-PLC wird kein Passwort benötigt.

IsSecureCommunication

Alle Konfigurationen haben das folgende Attribut: `IsSecureCommunication`. Das Attribut liefert "TRUE" im Falle einer "Sicheren Kommunikations"-Verbindung (TLS-Handshake) zur PLC. In allen anderen Fällen liefert das Attribut "FALSE". Weitere Informationen über `IsSecureCommunication` siehe Unterstützung der sicheren S7-Kommunikation mit TLS (Seite 492)

```
bool isSecureCommunication = moduleReadAccessPassword.IsSecureCommunication;
if (isSecureCommunication == true)
{
    // Secure communication with (Tls handshake)
    ...
}
```

Eine nicht behandelte Konfiguration, die den Ladevorgang verhindern kann, verursacht eine `EngineeringTargetInvocationException` und bricht den Ladevorgang ab.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Bei einer nicht behandelten Ausnahme im Delegate wird eine `EngineeringDelegateInvocationException` ausgelöst.

Beispiel für die Implementierung von `PreUploadDelegate`:

```
private static void PreConfigureUpload(UploadConfiguration UploadConfiguration)
{
    ModuleReadAccessPassword moduleReadAccessPassword = UploadConfiguration as
    ModuleReadAccessPassword;
    if (moduleReadAccessPassword != null)
    {
        string passWD = "passWD";
        var password = new SecureString();
        foreach (var c in passWD)
            password.AppendChar(c);
        moduleReadAccessPassword.SetPassword(password);
        return;
    }
    ModuleWriteAccessPassword moduleWriteAccessPassword = UploadConfiguration as
    ModuleWriteAccessPassword;
    if (moduleWriteAccessPassword != null)
    {
        string passWD = "passWD";
        var password = new SecureString();
        foreach (var c in passWD)
            password.AppendChar(c);
        moduleWriteAccessPassword.SetPassword(password);
        return;
    }
    ...
    throw new NotSupportedException(); // Exception thrown in the delagate will cancel upload
}
```

Parameter3: UploadOptions

Sie können die Upload-Optionen nicht angeben. Die Upload-Optionen lauten: "Hardware", "Software", "Hardware und Software" sowie "Hardware, Software und Dateien".

Der `StationUpload` arbeitet automatisch mit den Upload-Optionen "Hardware, Software und Dateien".

Upload über Router

TIA Portal Openness unterstützt den Stationsupload über einen Router, da die Station hinter dem Router nicht in der Projektnavigation sichtbar ist. Für den Upload über einen NAT-Router muss diese Adresse mit einer zugeordneten Adresse konfiguriert werden.

Der Upload über einen Router mit aktiviertem IP-Routing ist ebenfalls möglich.

Upload über Router mit IP-Routing

Das Gerät im Netzwerk hinter dem Router ist nur erreichbar, wenn der Router IP-Routing oder DNAT-Routing unterstützt. Anders als bei einem NAT-Router ist der Upload der PLC auch dann möglich, wenn nicht direkt auf die PLC zugegriffen werden kann, sondern über einen Kommunikationsprozessor (CP) oder ein Kommunikationsmodul (CM).

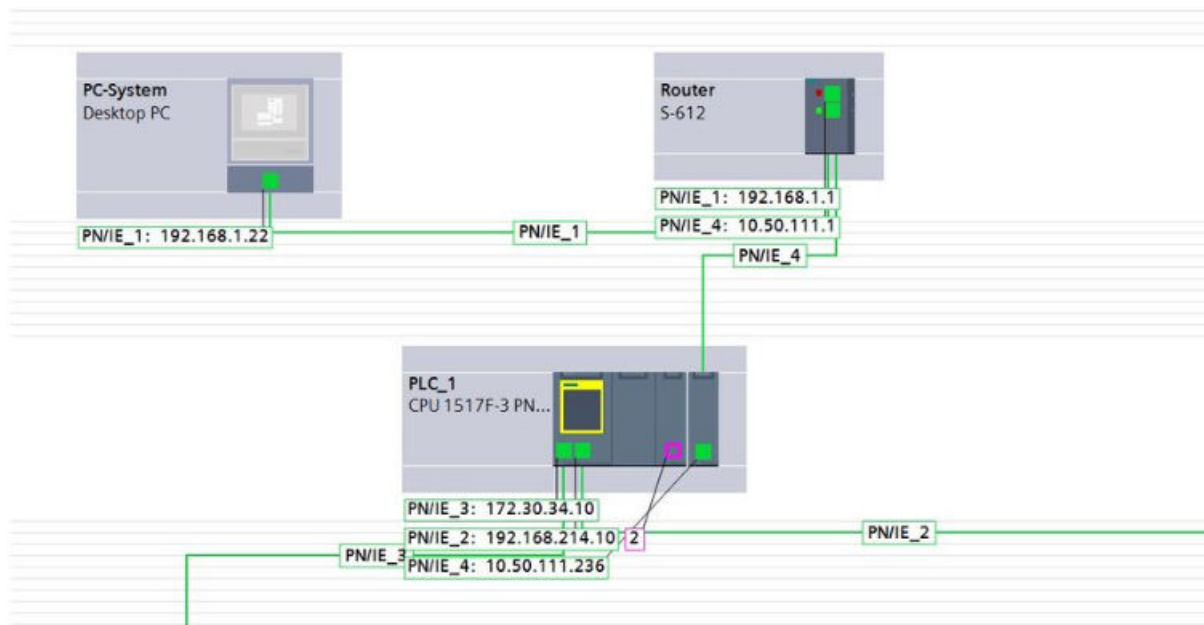
Daher werden folgende Szenarien unterstützt:

- Verbindung zwischen PG/PC und direkt mit dem Zielsubnetz verbundener Plus-PLC in Kombination mit einem Router mit IP-Routing.
- Verbindung zwischen PG/PC und direkt mit dem Zielsubnetz verbundenem Plus-PLC in Kombination mit einem Router mit DNAT-Routing.
- Verbindung zwischen PG/PC und Plus-PLC über ein Kommunikationsmodul (CM) in Kombination mit einem Router mit IP-Routing.
- Verbindung zwischen PG/PC und Plus-PLC über ein Kommunikationsmodul (CM) in Kombination mit einem Router mit DNAT-Routing.
- Verbindung zwischen PG/PC und Plus-PLC über einen Kommunikationsprozessor (CP) in Kombination mit einem Router mit IP-Routing.
- Verbindung zwischen PG/PC und Plus-PLC über einen Kommunikationsprozessor (CP) in Kombination mit einem Router mit DNAT-Routing.

Hinweis

Der Upload einer PLC 1200 V1 über ein CM oder einen CP wird nicht unterstützt; unterstützt wird lediglich das OMS+ Protokoll.

Beispiel: Ein Gerät befindet sich in einem privaten Subnetz. Die Adresse des Geräts in diesem Subnetz lautet 192.168.214.10. Ihr PG/PC befindet sich mit der Adresse 192.168.1.22 in einem anderen Subnetz. Zwischen den beiden Subnetzen wird ein Router mit IP-Routing eingesetzt. Der Upload des Geräts kann jetzt über die Adresse 10.50.111.236 des CPs erfolgen.

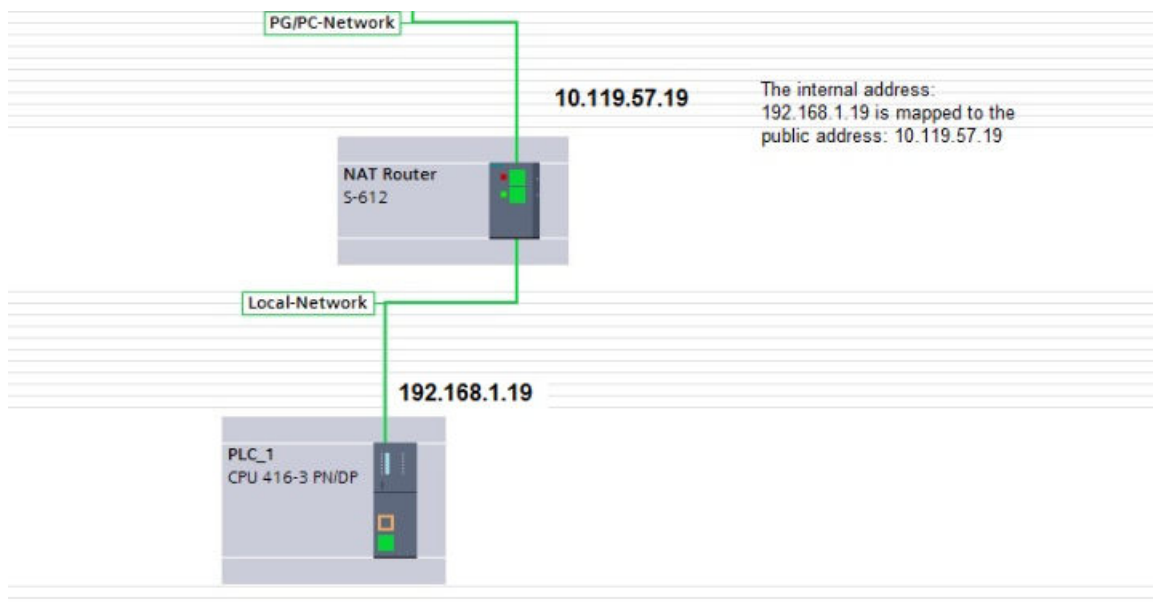


Upload über NAT-Router

Ein Gerät in einem privaten Netzwerk ist nur über einen NAT-Router erreichbar, wenn die IP-Adresse des Geräts direkt einer öffentlichen IP-Adresse zugeordnet wird (Destination-NAT). Insbesondere NAPT (Network Address Port Translation) wird nicht unterstützt.

Beispiel: Ein Gerät befindet sich in einem privaten Subnetz. Die Adresse des Geräts in diesem Subnetz lautet 192.168.1.19. Ihr PG/PC hat die Adresse 10.119.57.153 und befindet sich in einem anderen Subnetz. Zwischen den beiden Subnetzen wird ein NAT-Router eingesetzt. Der NAT-Router muss konfiguriert werden; dabei wird die Geräteadresse 192.168.1.19 einer gültigen Adresse zugeordnet, die sich im selben Subnetz wie der PG/PC befindet. Beispiel: 10.119.57.19.

Der Upload des Geräts kann jetzt über die Adresse 10.119.57.19 erfolgen (siehe Abschnitt **"Konfigurationsadresse"**).



Beim Upload über NAT-Router gelten einige wenige Einschränkungen.

Zunächst kann die Kommunikation über NAT-Router und über S7-Routing nicht zusammen verwendet werden. Das Gerät muss über seine IP-Adresse direkt vom NAT-Router aus erreichbar sein. An der Kommunikation zwischen NAT-Router und Gerät

darf kein S7-Router beteiligt sein. Deshalb können die Geräte S71500 intern und S7300 intern nicht vom PG/PC-System aus erreicht werden.

Das Engineering-System muss direkt über seine IP-Adresse auf den NAT-Router zugreifen können. An der Kommunikation zwischen Engineering-System und NAT-Router darf kein S7-Router beteiligt sein. Deshalb kann das

Engineering-System PG/PC 2 in der obigen Abbildung nicht zum Upload eines Geräts aus dem Testnetzwerk verwendet werden.

UploadResult

Das von der Upload-Aktion zurückgegebene Ergebnis UploadResult liefert Rückmeldung über den Zustand der geladenen Objekte.

- UploadResult.Message: UploadResultMessageComposition - Zusammensetzung von UploadResultMessage

Die folgenden Attribute werden unterstützt:

Attribute	Beschreibung
ErrorCount	int-Wert der Fehler bei hergestellter Verbindung zwischen PG/PC und S7-1500 PLC über ein Kommunikationsmodul (CM) in Kombination mit einem Router mit IP-Routing.
State	UploadResultState mit den möglichen Werten: Success, Information, Warning und Error
UploadedStation	Eine Device-Instanz der geladenen Station
WarningCount	Anzahl der Warnungen beim Laden als int

Die UploadResultMessage enthält:

- UploadResultMessage.Messages : UploadResultMessageComposition - Zusammensetzung von UploadResultMessage

Die folgenden Attribute werden unterstützt:

Attribute	Beschreibung
DateTime	System.DateTime der erstellten Meldung.
ErrorCount	Ein int-Zähler für Fehler.
State	UploadResultState mit den möglichen Werten: Success, Information, Warning und Error
WarningCount	Anzahl der Warnungen beim Laden als int

Beispiel für den Aufruf von Upload

```
internal bool UploadPLC()
{
    ...
    UploadResult result = uploadProvider.StationUpload(uploadAddress, preUploadDelegate);
    ...
    PrintAllMessages(result.Messages, 0);
    ...
}
internal void PrintAllMessages(UploadResultMessageComposition messages, int level)
{
    if (messages == null)
        return;

    if (level == 0)
        Console.WriteLine("\n");
    foreach (UploadResultMessage message in messages)
    {
        string messageOut = message.Message.PadLeft(message.Message.Length + level, '\t') + "\n";
        Console.WriteLine(messageOut);
        if ((message.Messages != null) && (message.Messages.Count > 0))
            PrintAllMessages(message.Messages, level+1);
    }
}
```

5.11.1.13 PLC-Software vergleichen**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie haben die folgenden Möglichkeiten, um die Abweichung zwischen der Software zweier Geräte zu ermitteln:

- Vergleich der Software zweier konfigurierter PLCs
- Vergleich der Software eines PLC und der Projektbibliothek
- Vergleich der Software eines PLC und der globalen Bibliothek
- Vergleich der Software eines PLC und der Masterkopie eines PLC
- Vergleich der Software eines konfigurierten PLC mit der Software eines verbundenen PLC im Status „online“

Signatur

Verwenden Sie die Methode `CompareTo` oder `CompareToOnline` für den Vergleich.

```
public CompareResult CompareTo (ISoftwareCompareTarget
compareTarget)
```

```
public CompareResult CompareToOnline ()
```

Rückgabewert/Parameter	Funktion
<code>CompareResult compareResult</code>	Gibt das Vergleichsergebnis zurück: • <code>FolderContentsDifferent</code>: Inhalt der verglichenen Ordner unterscheidet sich. • <code>FolderContentsIdentical</code>: Inhalt der verglichenen Ordner ist identisch. • <code>ObjectsDifferent</code>: Inhalt der verglichenen Objekte unterscheidet sich. • <code>ObjectsIdentical</code>: Inhalt der verglichenen Objekte ist identisch. • <code>LeftMissing</code>: Das Objekt ist nicht in dem Objekt enthalten, von dem aus der Vergleich gestartet wurde. • <code>RightMissing</code>: Das Objekt ist nicht in dem Objekt enthalten, das verglichen wird. • <code>CompareIrrelevant</code>: Der Vergleich dieser 2 Objekte ist irrelevant. • <code>FolderContainsDifferencesOwnStateDifferent</code>: Der Inhalt des Ordners weist einen oder zwei Unterschiede auf, der eigene Zustand des Ordners weicht ab. • <code>FolderContentEqualOwnStateDifferent</code>: Der Ordnerinhalt ist identisch, der eigene Zustand des Ordners weicht ab.
<code>ISoftwareCompareTarget compareTarget</code>	Liste vergleichbarer Objekte.

Programmcode

Ändern Sie den folgenden Programmcode, um das Vergleichsergebnis auszugeben:

```
private static void WriteResult(CompareResultElement compareResultElement, string indent)
{
    Console.WriteLine("{0} <{1}> <{2}> <{3}> <{4}> ",
        indent,
        compareResultElement.LeftName,
        compareResultElement.ComparisonResult,
        compareResultElement.RightName,
        compareResultElement.DetailedInformation);
    WriteResult(compareResultElement.Elements, indent);
}
private static void WriteResult (IEnumerable<CompareResultElement> compareResultElements,
string indent)
{
    indent += " ";
    foreach (CompareResultElement compareResultElement in compareResultElements)
    {
        WriteResult(compareResultElement, indent);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software von Geräten zu vergleichen:

```
private static void CompareTwoOfflinePlcs(PlcSoftware plcSoftware0, PlcSoftware
plcSoftware1)
{
    if (plcSoftware0 != null && plcSoftware1 != null)
    {
        CompareResult compareResult = plcSoftware0.CompareTo(plcSoftware1);
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software eines PLC mit der Projektbibliothek zu vergleichen:

```
private static void ComparePlcToProjectLibrary(Project project, PlcSoftware plcSoftware)
{
    if (project != null && plcSoftware != null)
    {
        CompareResult compareResult = plcSoftware.CompareTo(project.ProjectLibrary);
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```


Ändern Sie den folgenden Programmcode, um die Software eines PLC mit der globalen Bibliothek zu vergleichen:

```
private static void ComparePlcToGlobalLibrary(PlcSoftware plcSoftware, GlobalLibrary
globalLibrary)
{
    if (plcSoftware != null && globalLibrary != null)
    {
        CompareResult compareResult = plcSoftware.CompareTo(globalLibrary);
        WriteResult(compareResult.RootElement, String.Empty);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software eines PLC mit einer Masterkopie zu vergleichen:

```
private static void ComparePlcToMasterCopy(Project project, PlcSoftware plcSoftware)
{
    if (project != null && plcSoftware != null)
    {
        CompareResult compareResult =
plcSoftware.CompareTo(project.ProjectLibrary.MasterCopyFolder.MasterCopies[0]);
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software eines PLC mit der Software eines verbundenen PLC zu vergleichen:

```
private static void ComparePlcToOnlinePlc(PlcSoftware plcSoftware)
{
    if (plcSoftware != null)
    {
        CompareResult compareResult = plcSoftware.CompareToOnline();
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

5.11.1.14 PLC-Programm aktualisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit `UpdateProgram()` eine Aktualisierung auf die neueste Version der PLC vornehmen. Auf `UpdateProgram()` können Sie über die `PlcSoftware`-Instanz zugreifen.

Programmcode

```
private void UpdateProgramPLC()  
{  
    PlcSoftware plcSoftware = ...;  
    plcSoftware.UpdateProgram();  
}
```

Siehe auch

Projekt öffnen (Seite 131)

5.11.1.15 PLC-Hardware vergleichen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet. Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Mithilfe von TIA Portal Openness können Sie die Hardware zweier PLC-Geräte vergleichen.

Signatur

Verwenden Sie die Methode `CompareTo` für den Vergleich der beiden Hardwareobjekte.

`CompareResult CompareTo (IHardwareCompareTarget compareTarget);`

Rückgabewert/-parameter	Funktion
<code>CompareResult compare result</code>	Das Vergleichsergebnis zurückgeben: • <code>FolderContainsDifferencesOwnStateDifferent</code>: Der Inhalt des Ordners weist einen oder zwei Unterschiede auf, der eigene Zustand des Ordners weicht ab. • <code>FolderContentEqualOwnStateDifferent</code>: Der Ordnerinhalt ist identisch, der eigene Zustand des Ordners weicht ab.
<code>IHardwareCompareTarget compareTarget</code>	Das Vergleichsziel, wegen dessen der Hardwarevergleich durchgeführt wird. Darf nicht null sein.

Wenn der Parameter `compareTarget` null ist und ein Hardwarevergleich versucht wird, wird die `Siemens.Engineering.EngineeringTargetInvocationExceptions` ausgelöst.

Programm

Ändern Sie den folgenden Programmcode, um das Vergleichsergebnis auszugeben:

```
...
CompareResult compareResult = plc_1.CompareTo(plc_2);
CompareResultState resultState = compareResult.RootElement.ComparisonResult;
if (resultState == CompareResultState.FolderContainsDifferencesOwnStateDifferent)
{
    // Folder contents have one or more differences, folder's own state is different:
    // May occur if the plc has a different subordinate element, e.g., a local module, and
    // the plc itself is different, e.g., in a parameter
}
else if (resultState == CompareResultState.FolderContentEqualOwnStateDifferent)
{
    // Folder content is the same, folder's own state is different:
    // May occur if a folder-style module, e.g., FM 351, has equal subordinate elements but
    // the module itself is different, e.g., in a parameter
}
else if (resultState == CompareResultState.FolderContentsIdentical)
{
    ...
}
...
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.1.16 Passwortrichtlinie für PLC festlegen

Voraussetzung

- TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness eine Sicherheitspasswortrichtlinie für Legacy-PLCs und Plus-PLCs festlegen.

Passwortrichtlinie für Legacy-PLC festlegen

Mit LegacyPlcPasswordPolicyService am Objekt ProjectBase können Sie eine Sicherheitspasswortrichtlinie für Legacy-PLCs festlegen.

Die Funktion LegacyPlcPasswordPolicyService unterstützt die folgenden Attribute:

Attributname	Datentyp	Beschreibung
PasswordPolicyEnabled	Boolean	Wenn PasswordPolicyEnabled "True" ist, werden beim Überprüfen des Passworts alle anderen konfigurierten Attributwerte berücksichtigt. Wenn PasswordPolicyEnabled "False" ist, kann die neue Richtlinie nicht auf Legacy-PLCs angewendet werden.
MinimumLength	Short	Legt den Wert für minimumLength des Passworts fest Der zulässige Bereich für die Mindestdlänge liegt zwischen 5 und 8
MinimumNumericCharacterLength	Short	Legt den Wert für minimumNumericCharacterLength im Passwort fest Der zulässige Bereich für die Mindestdlänge numerischer Zeichen liegt zwischen 0 und 8
MinimumSpecialCharacterLength	Short	Legt den Wert für minimumSpecialCharacterLength im Passwort fest Der zulässige Bereich für die Mindestdlänge numerischer Zeichen liegt zwischen 0 und 8
IncludesLowerCaseAndUpperCaseCharacters	Boolean	Legt den Wert für includesLowerCaseAndUpperCaseCharacters im Passwort fest

Beim Versuch, einen Attributwert außerhalb des zulässigen Bereichs festzulegen, wird eine benutzerdefinierte Ausnahme PasswordPolicySettingsException ausgelöst.

Passwortrichtlinie für S7-1500 PLC festlegen

Mit PlcPasswordPolicyService am Objekt ProjectBase können Sie eine Sicherheitspasswortrichtlinie für eine Plus-PLC festlegen.

Die Funktion PlcPasswordPolicyService unterstützt die folgenden Attribute:

Attributname	Datentyp	Beschreibung
PasswordPolicyEnabled	Boolean	Wenn PasswordPolicyEnabled "True" ist, werden beim Überprüfen des Passworts die für UMAC (d.h. Passwortkomplexität für Projektbenutzer) konfigurierten Attributwerte berücksichtigt. Wenn PasswordPolicyEnabled "False" ist, kann die neue Richtlinie nicht auf S7-1500 PLCs angewendet werden.

Ausnahme

Wenn das Passwort mit dem Openness-Skript für eine bestimmte Schutzstufe festgelegt wird, wird eine neue Art von Ausnahme EngineeringPasswordPolicyViolationException ausgelöst, falls das eingegebene Passwort nicht der Richtlinie entspricht.

Programmcode

Ändern Sie folgenden Programmcode, um die Passwortrichtlinien für Legacy-PLCs festzulegen:

```
public void LegacyPlc_SecuritySettingPasswordPolicy_SetAttributes()
{
    LegacyPlcPasswordPolicyService m_LegacyPlcPasswordPolicyService =
    m_project.GetService<LegacyPlcPasswordPolicyService>();
    m_LegacyPlcPasswordPolicyService.PasswordPolicyEnabled= true;
    m_LegacyPlcPasswordPolicyService.MinimumLength = 6;
    m_LegacyPlcPasswordPolicyService.MinimumNumericCharacterLength = 2;
    m_LegacyPlcPasswordPolicyService.MinimumSpecialCharacterLength=2;
    m_LegacyPlcPasswordPolicyService.IncludeLowerCaseAndUpperCaseCharacters = true;
}
```

Ändern Sie den folgenden Programmcode, um die Passwortrichtlinien für S7-1500 PLC festzulegen:

```
public void PlusPlc_SecuritySettingPasswordPolicy_SetAttributes()
{
    PlcPasswordPolicyService m_PlacPasswordPolicyService =
    m_project.GetService<PlcPasswordPolicyService>();
    m_PlusPlcPasswordPolicyService.PasswordPolicyEnabled= true;
}
```

5.11.1.17 Verwalten von dynamischen Zertifikatseinstellungen

Voraussetzung

- TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie das dynamische Zertifikatsmanagement für alle unterstützten Dienste an einer zentralen Stelle im TIA Portal konfigurieren, anstatt jede Anwendung und jeden Dienst einzeln zu ändern.

Sie können den Dienst Certificate Management Configuration zur Verwaltung eines dynamischen Zertifikats nutzen, das für die PLC-Instanz für alle PLCs der Gerätefamilie ab Version V3.0 verfügbar ist.

Eigenschaft

CertificateManagementConfiguration unterstützt die folgenden Eigenschaften:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff	Hinweis
Usage	CertificateConfigurationUsage	Empfängt oder gibt die Verwendung der Konfiguration der Zertifikatsverwaltung an Mit der Eigenschaft Usage können Sie zwischen zwei Optionen wählen: entweder Zertifikate, die über TIA Portal konfiguriert und heruntergeladen werden, oder Zertifikate, die von der Zertifikatsverwaltung während der Laufzeit bereitgestellt werden. Die Eigenschaft Usage hat zwei Enum-Werte: • TIAPortal • Runtime	Lesen/ Schreiben	EngineeringTargetInvocationException wird bei einer eingestellten Funktion für den Wert CertificateConfigurationUsage.Runtime ausgelöst, wenn WebServer und OpcUaServer deaktiviert sind
CertificateExpirationEventActivated	Boolean	Aktivieren oder Deaktivieren der Einstellung für den Ablauf von Zertifikaten in der Konfiguration der Zertifikatsverwaltung	Lesen/ Schreiben	EngineeringTargetInvocationException wird ausgelöst, wenn die Eigenschaft Usage als konfigurierte Zertifikate und Download über TIA Portal eingestellt ist

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff	Hinweis
RemainingCertificateLifetime	ushort	Empfängt oder gibt den Wert für die Anzeige des Ereignisses für die verbleibende Zertifikatslebensdauer der Konfiguration der Zertifikatsverwaltung an Das ist ein prozentualer Wert, mit dem die Zeit berechnet werden kann, nach der ein PLC-Alarm ausgelöst werden soll, um den Benutzer über einen bevorstehenden Ablauf zu informieren.	Lesen/ Schreiben	EngineeringTargetInvocationException wird ausgelöst, wenn der Eigenschaftswert EnableCertificateExpirationEvent auf Falsch oder der Wert nicht zwischen 10 – 90 eingestellt ist
CertificateSupportedServices	CertificateSupportedServiceComposition	Stellt alle von dem Zertifikat unterstützten Dienste der Konfiguration der Zertifikatsverwaltung dar, die die dynamische Zertifikatsverwaltung aktiviert haben. Die momentan unterstützten Dienste sind OPC UA und Webserver. Die momentan unterstützten Dienste sind OPC UA und Webserver.	Lesen	

Methode

CertificateSupportedService() stellt die unterstützten Dienste der Konfiguration der Zertifikatsverwaltung dar. Akzeptiert die folgende Eigenschaft:

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff	Hinweis
ServiceType	CertificateSupportedServiceName	Stellt den Namen des Diensts dar, der von der Konfiguration der Zertifikatsverwaltung unterstützt wird. CertificateSupportedServiceName akzeptiert das folgende Enum-Element und seinen Enum-Wert: • Webserver - 100 • OpcUaServer - 200	Lesen	
Id	ushort	Stellt die ID des Diensts dar, der von der Konfiguration der Zertifikatsverwaltung unterstützt wird.	Lesen	
ServiceGroupName	string	Stellt den Gruppennamen der unterstützten Dienste der Konfiguration der Zertifikatsverwaltung dar	Lesen/ Schreiben	EngineeringTargetInvocationException wird ausgelöst, wenn der Wert des Diensts auf mehr als 64 Zeichen eingestellt ist.

Programmcode

```
//...
// for getting the service
CertificateManagementConfiguration certificateConfigurationService =
plc.GetService<CertificateManagementConfiguration>();
// To setup the usage
certificateConfigurationService.Usage = CertificateConfigurationUsage.TIAPortal;
// for service count
var servicesCount = certificateConfigurationService.CertificateSupportedServices.Count;
// For setting group name
certificateConfigurationService.CertificateSupportedServices[0].ServiceGroupName =
"ServiceName_1";
```

Umgang mit dynamischen Zertifikaten während des Ladens in die PLC

SelectiveDeleteDownload kann verwendet werden, um Zertifikate während des Ladens in die PLC zu verarbeiten. Basierend auf dieser Option werden dynamische Zertifikate gelöscht.

SelectiveDeleteDownload.CurrentSelection unterstützt das folgende Enum-Element und seinen Enum-Wert:

- SelectiveDeleteDataSelections.AcceptAll - alle akzeptieren
- SelectiveDeleteDataSelections.DeleteSelected - Auswahl löschen
- SelectiveDeleteDataSelections.DeleteAll - alle löschen

Programmcode: Umgang mit dynamischen Zertifikaten während das Ladens in die PLC

```
private void DownloadConfigurationDelegate(DownloadConfiguration downloadConfiguration)
{
    if (downloadConfiguration is SelectiveDeleteDownload selectiveDeleteDownload)
    {
        selectiveDeleteDownload.CurrentSelection = SelectiveDeleteDataSelections.AcceptAll;
    }
}
```

5.11.2 Funktionen für den Zugriff auf den PLC-Dienst

5.11.2.1 Einstellung der Zugriffsstufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Mit der TIA Portal Openness API können Sie über die Hardware PlcAccessLevelProvider an der DeviceItem CPU des Geräts auf die Einstellung der Zugriffsstufen zugreifen.

Diese Funktion bietet Aktionen zum Festlegen/Zurücksetzen des Passworts und ein modelliertes Attribut, um die Einstellung der Zugriffsstufen festzulegen/auszulesen.

- Void SetPassword (accessLevelType PlcProtectionAccessLevel, Passwort SecureString)
- Void ResetPassword (accessLevelType PlcProtectionAccessLevel)
- Attribut PlcProtectionAccessLevel

Im Allgemeinen ist die Handhabung über Openness ähnlich wie die in der TIA Portal UI.

- PlcProtectionAccessLevel kann an S7-1200/1500 PLCs immer ausgelesen und festgelegt werden.
- Wenn für die Zugriffsstufe eine andere Einstellung als Vollzugriff festgelegt ist (oder Vollzugriff inkl. Fail-safe), muss ein Passwort für Vollzugriff eingerichtet werden. Ansonsten kommt es zu einem Übersetzungsfehler: "Passwort darf nicht leer sein"
- Es ist nicht zulässig, das gleiche Passwort für verschiedene Zugriffsstufen festzulegen.
- Es können nur Passwörter für höhere Stufen als die festgelegte Zugriffsstufe eingerichtet / zurückgesetzt werden (sofern für PlcProtectionAccessLevel HMIAccess eingestellt ist, kann nur für ReadAccess und FullAccess ein Passwort eingerichtet werden).

Mögliche Ausnahmen:

- EngineeringTargetInvocationException: **"Gleiches Passwort ist bereits für andere Zugriffsstufe vorhanden"** – wird zurückgegeben, wenn der Benutzer versucht, das gleiche Passwort für unterschiedliche Zugriffsstufen festzulegen.
- EngineeringTargetInvocationException: **"Passwort kann für Zugriffsstufe nicht festgelegt werden"** – wird zurückgegeben, wenn der Benutzer versucht, ein Passwort für eine strengere Zugriffsstufe als die ausgewählte festzulegen.
- EngineeringTargetInvocationException: **"Passwort darf nicht leer sein"** – wird zurückgegeben, wenn der Benutzer versucht, ein leeres Passwort festzulegen.
- EngineeringTargetInvocationException: **"Zugriffsstufe ist nicht gültig"** – wird zurückgegeben, wenn der Benutzer versucht, an nicht fehlersicheren PLCs für die Zugriffsstufe FullAccessIncludingFailsafe festzulegen.

Zugriffsstufe

In Openness ist die Zugriffsstufe als Enumeration mit dem Namen PlcProtectionAccessLevel modelliert. Der Name der Enumerationsfelder basiert auf den Einträgen "Access Level" für S7-1200/1500 PLCs.

Name im TIA UI	Enumerations-Eintrag	Wert	Anmerkungen
-	None	0	Für Initialisierung der Enumeration. Dieser Wert darf nicht vom Openness-Benutzer festgelegt werden.
Vollzugriff (kein Schutz)	FullAccess	1	
Lesezugriff	ReadAccess	2	
HMI-Zugriff	HMIAccess	3	
Kein Zugriff	NoAccess	4	
Vollzugriff inkl. Fail-safe (kein Schutz)	FullAccessIncludingFailsafe	5	Nur bei fehlersicheren PLCs verfügbar.

Bei PLCs ist ein EOM-Attribut (modelliertes Attribut) mit dem Namen PlcProtectionAccessLevel vom Typ PlcProtectionAccessLevel-Enumeration verfügbar, um die Zugriffsstufe festzulegen/abzurufen.

Sie können das folgende Codebeispiel für PlcProtectionAccessLevel für "Kein Zugriff" verwenden:

```
DeviceItem S71500PLC = ...;
PlcAccessLevelProvider myPlcAccessLevelProvider =
S71500PLC.GetService<PlcAccessLevelProvider>();
myPlcAccessLevelProvider.PlcProtectionAccessLevel = PlcProtectionAccessLevel.NoAccess;
```

Passwort

Das Passwort ist aus Sicherheitsgründen als `SecureString` implementiert. Die Passwörter für die verschiedenen Zugriffsstufen können nur mit der entsprechenden Aktion festgelegt/zurückgesetzt werden. Das Auslesen der Passwörter wird nicht unterstützt.

```
myPlcAccessLevelProvider.SetPassword(PlcProtectionAccessLevel.ReadAccess,
someSecureString);
myPlcAccessLevelProvider.ResetPassword(PlcProtectionAccessLevel.ReadAccess);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

5.11.2.2 Auf Software-Prüfsumme zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Mit der Openness API können Sie auf die Software-Prüfsumme einer PLC-Station zugreifen.

Programmcode

Für den Zugriff rufen Sie die Instanz von `PlcSoftware` ab und laden eine Dienstinstantz von `PlcChecksumProvider`.

```
PlcSoftware plc = ...;
//get PlcSoftware instance
PlcChecksumProvider checksumProvider =
plc.GetService<PlcChecksumProvider>();
```

Falls der PLC die Prüfsummenberechnung nicht unterstützt, gibt der `GetService` -Aufruf null zurück.

An der `PlcChecksumProvider`-Instanz kann die Software-Prüfsumme wie folgt erreicht werden:

```
string softwareChecksum = checksumProvider.Software;
```

Das `Software`-Attribut ist schreibgeschützt und es gibt null zurück, wenn das Programm nicht übersetzt ist.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.2.3 PC-Schnittstelle zuordnen**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Für die Schnittstellenzuordnung müssen Sie den `PcInterfaceAssignment`-Dienst von `Siemens.HW.Features` auf der Geräteelementschnittstelle abrufen.

Wenn die PC-Station eine Version ≤ 2.0 hat, geben die Methoden dieses Diensts null zurück und die Schnittstellenzuordnung ist nicht möglich bei Schnittstellen, die nur der PC-Station oder nur Windows oder keinem von beiden zugeordnet sind. Bei Schnittstellen, die einer SW-CPU zugeordnet sind, ist dies wieder möglich.

Hinweis

Beim CAX-Austausch ist die Unterstützung der PC-Station sehr begrenzt. Es werden nur PC-Stationen mit Version 1.0 und dem Standardwert für `InterfaceAssignment` unterstützt. Das liegt daran, dass diese beiden Werte sich auf die Gerätestruktur auswirken (d. h. was eingebaut ist, was wo gesteckt werden kann). Da jedoch die Parameter nicht zur Datei `AutomationML` gehören, werden die Werte beim Import nicht gesetzt. Folglich ist die Struktur des IPC nicht passend und die anderen Module können nicht gesteckt werden.

Programmcode: Schnittstellenzuordnung

Für die Schnittstellenzuordnung ändern und verwenden Sie folgenden Programmcode:

```
DeviceItem swCpu = ...;
DeviceItem myInterface = ...;
PcInterfaceAssignment provider = myInterface.GetService<PcInterfaceAssignment>();
//user has to give device item sw cpu in case he/she wants to assign to it. (in the future,
there could be multi sw-cpu cases)
provider.AssignInterface(PcInterfaceAssignmentMode.SoftwarePlc, swCpu);
...
provider.AssignInterface(PcInterfaceAssignmentMode.PcStation);
...
provider.AssignInterface(PcInterfaceAssignmentMode.None);
...
```

Um Informationen über den aktuellen Zuordnungsmodus einer Schnittstelle abzurufen, ändern und verwenden Sie folgenden Programmcode:

```
PcInterfaceAssignment provider = myInterface.GetService<PcInterfaceAssignment>();
PcInterfaceAssignmentMode mode = provider.PcInterfaceAssignmentMode;
switch(mode)
{
case PcInterfaceAssignmentMode.None: Console.WriteLine("Assigned to none or windows-
only.");
break;
case PcInterfaceAssignmentMode.PcStation: Console.WriteLine("Assigned to pc-station.");
break;
case PcInterfaceAssignmentMode.SoftwarePlc: Console.WriteLine("Assigned to SoftwarePlc: "
+ provider.SoftwarePlc.Name);
break;
}
```

Programmcode: Konfiguration von Hardware-Ressource und IPC-Erweiterung

Mit dem folgenden Code können Sie auch die richtige Eigenschaft 'IPC-Erweiterung und Hardware-Ressource' über den PcInterfaceAssignment-Dienst auswählen:

```
DeviceItem myInterface = ...;
PcInterfaceAssignment provider = myInterface.GetService<PcInterfaceAssignment>();
//this method returns a subset of IpcExpansion values which are available for the IPC that
the interface is plugged in. IEnumerable<string> ipcExpansionChoices =
provider.GetAvailableIPCExpansions();
string myChoice;
foreach(var choice in ipcExpansionChoices)
{
//user must select the desired value depending on his/her configuration
if (choice.Contains("3yxx1")
{
myChoice = choice;
break;
}
}
provider.IpcExpansion = myChoice;
//After assigning IpcExpansion value, user may assign HardwareResource value using the
following enumeration.
provider.HardwareResource = HardwareResource.X101;
//OR
//myInterface.SetAttribute("HardwareResource", HardwareResource.X101);
//myInterface.SetAttribute("IpcExpansion", mychoice);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.2.4 Status einer PLC ermitteln

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet. Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können den Zustand einer PLC oder aller PLCs in einem Projekt ermitteln.

TIA Portal Openness unterscheidet zwischen folgenden Zuständen:

- Offline
- PLC ist verbunden („Verbindung wird hergestellt“)
- Online
- PLC ist nicht verbunden („Verbindung wird getrennt“)
- Inkompatibel
- Nicht erreichbar
- Geschützt

Programmcode

Ändern Sie folgenden Programmcode, um den Zustand eines PLC zu ermitteln:

```
public static OnlineState GetOnlineState(DeviceItem deviceItem)
{
    OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
    return onlineProvider.State;
}
```

Ändern Sie folgenden Programmcode, um den Zustand aller PLCs in einem Projekt zu ermitteln:

```
public static void DetermineOnlineStateOfAllProjectDevices(Project project)
{
    foreach (Device device in project.Devices)
    {
        foreach (DeviceItem deviceItem in device.DeviceItems)
        {
            OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
            if (onlineProvider != null)
            {
                OnlineState state = onlineProvider.State;
            }
        }
    }
}
```

5.11.2.5 Auf Parameter einer Online-Verbindung zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Parameter für eine Online-Verbindung ermitteln oder festlegen:

- Verfügbare Verbindungsarten mit einer PLC enumerieren
- Verfügbare Schnittstellen zu einer PLC enumerieren
- Zugeordnete Steckplätze enumerieren
- Verfügbare Adressen der Subnetze und Gateways enumerieren
- Verbindungsparameter festlegen

Für Informationen über IsSecureCommunication siehe Unterstützung der sicheren S7-Kommunikation TLS (Seite 492)

Programmcode: Verbindungsparameter ermitteln

Um die verfügbaren Verbindungsarten, PC-Schnittstellen und Steckplätze zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateConnectionModesOfPLC(DeviceItem deviceItem)
{
    OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
    if (onlineProvider == null)
    {
        return; // Only cpu device items can provide OnlineProvider service
    }
    // Accessing connection configuration object
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    // Now access connection configuration members
    foreach (ConfigurationMode mode in configuration.Modes)
    {
        Console.WriteLine("Mode name:{0}", mode.Name);
        foreach (ConfigurationPcInterface pcInterface in mode.PcInterfaces)
        {
            Console.WriteLine("PcInterface name:{0}", pcInterface.Name);
            Console.WriteLine("PcInterface number:{0}", pcInterface.Number);
            foreach (ConfigurationTargetInterface targetInterface in
pcInterface.TargetInterfaces)
            {
                Console.WriteLine("TargetInterface:{0}", targetInterface.Name);
            }
        }
    }
}
```

Sie können auch über den Namen auf eine Verbindungsart und eine PC-Schnittstelle zugreifen:

```
public static ConfigurationTargetInterface
GetTargetInterfaceForOnlineConnection(OnlineProvider onlineProvider)
{
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    ConfigurationMode mode = configuration.Modes.Find("PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("PLCSIM", 1);
    ConfigurationTargetInterface slot = pcInterface.TargetInterfaces.Find("2 X3");
    return slot;
}
```


Um die an einer PC-Schnittstelle verfügbaren Adressen von Subnetzen und Gateways zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumeratingPCInterfaceSubnetsAndGateways(ConfigurationPcInterface
pcInterface)
{
    foreach (ConfigurationSubnet subnet in pcInterface.Subnets)
    {
        Console.WriteLine("Subnet name:{0}", subnet.Name);
        foreach (ConfigurationGateway gateway in subnet.Gateways)
        {
            //Get the name of the gateway:
            Console.WriteLine("Gateway name:{0}", gateway.Name);
            //Get the IP address of each gateway:
            foreach (ConfigurationAddress gatewayAddress in gateway.Addresses)
            {
                Console.WriteLine("Gateway Address:{0} has {1}", gatewayAddress.Name,
gatewayAddress.Address);
            }
        }
    }
}
```

Sie können auch über den Namen oder die IP-Adresse auf Subnetze und Gateways zugreifen:

```
public static void AccessSubnetAndGatewayOfPCInterface(ConfigurationPcInterface
pcInterface)
{
    ConfigurationSubnet subnet = pcInterface.Subnets.Find("PN/IE_1");
    ConfigurationAddress subnetAddress = subnet.Addresses.Find("192.168.0.1");
    ConfigurationGateway gateway = subnet.Gateways.Find("Gateway 1");
    ConfigurationAddress gatewayAddress = gateway.Addresses.Find("192.168.0.2");
}
```

Programmcode: Verbindungsparameter festlegen

Hinweis

Durch Festlegen der Verbindungsparameter werden alle zuvor festgelegten Verbindungsparameter überschrieben. Wenn Sie die Verbindungsparameter bereits direkt im TIA Portal festgelegt haben, brauchen Sie `ApplyConfiguration` nicht aufzurufen. Besteht bereits eine Online-Verbindung mit einem PLC, während `ApplyConfiguration` aufgerufen wird, wird eine Ausnahme ausgelöst.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie den folgenden Programmcode, um Steckplatzparameter festzulegen:

```
public static void SetConnectionWithSlot(OnlineProvider onlineProvider)
{
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    ConfigurationMode mode = configuration.Modes.Find(@"PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("PLCSIM", 1);
    // or network pc interface that is connected to plc
    ConfigurationTargetInterface slot = pcInterface.TargetInterfaces.Find("2 X3");
    configuration.ApplyConfiguration(slot);
    // After applying configuration, you can go online
    onlineProvider.GoOnline();
}
```

Ändern Sie den folgenden Programmcode, um Gateway-Adressparameter einzustellen:

```
public static void SetConnectionWithGatewayAddress(OnlineProvider onlineProvider, string
subnetName, string gatewayAddressName)
{
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    ConfigurationMode mode = configuration.Modes.Find(@"PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("PLCSIM", 1);
    // or network pc interface that is connected to plc
    ConfigurationSubnet subnet = pcInterface.Subnets.Find(subnetName);
    ConfigurationAddress gatewayAddress = subnet.Addresses.Find(gatewayAddressName);
    configuration.ApplyConfiguration(gatewayAddress);
    // After applying configuration, you can go online
    onlineProvider.GoOnline();
}
```

Ändern Sie den folgenden Programmcode, um Subnetz-Adressparameter einzustellen:

```
public static void SetConnectionWithSubnetAddress(OnlineProvider onlineProvider, string
subnetName)
{
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    ConfigurationMode mode = configuration.Modes.Find(@"PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("PLCSIM", 1);
    // or network pc interface that is connected to plc
    ConfigurationSubnet subnet = pcInterface.Subnets.Find(subnetName);
    ConfigurationAddressComposition addresses = subnet.Addresses;
    configuration.ApplyConfiguration(addresses[0]);
    // After applying configuration, you can go online
    onlineProvider.GoOnline();
}
```

Siehe auch

Unterstützung der sicheren S7-Kommunikation mit TLS (Seite 492)

5.11.2.6 Zugriff auf den Fingerabdruck zum schnellen Vergleich von Stationen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist offline.

Anwendung

Mit TIA Portal Openness können Sie FingerprintData für unterschiedliche Aspekte der PLC-Gerätekonfiguration abrufen, um einen schnellen Vergleich von Stationen durchzuführen. Hierzu verwenden Sie den Service FingerprintDataProvider, den Sie über ein vorgegebenes TIA Portal abrufen können. Über den Aufruf GetService wird eine Instanz von FingerprintDataProvider zurückgegeben. Andernfalls wird null zurückgegeben.

Programmcode: Abrufen von FingerprintDataProvider aus dem TIA Portal

```
TiaPortal tia = new TiaPortal(TiaPortalMode.WithUserInterface);
FingerprintDataProvider fingerprintDataProvider =
tia.GetService<FingerprintDataProvider>();
if (fingerprintDataProvider != null)
{
    ...
}
```

Parameter der Methode FingerprintDataProvider

Um die FingerprintData eines Geräts abzurufen, müssen Sie die Methode GetFingerprintData von FingerprintDataProvider aufrufen.

Im FingerprintDataProvider werden die folgenden Attribute unterstützt:

Parametername	Typ	Beschreibung
configurationAddress	Siemens.Engineering.Online.Configuration	Adresse des Geräts, für das die FingerprintData abgerufen werden sollen.
onlineConfigurationDelegate	Siemens.Engineering.Online	Delegate, der aufgerufen wird, um die Konfiguration vor dem Abrufen der FingerprintData zu prüfen.

Konfigurationsadresse

Sie müssen für `GetFingerprintData` ein Objekt `ConfigurationAddress` bereitstellen. Das Adressobjekt wird verwendet, um eine Verbindung zu dem Gerät aufzubauen, für das die `FingerprintData` abgerufen werden sollen. Das Objekt `ConfigurationAddress` muss in der `ConnectionConfiguration` des `FingerprintDataProvider` erstellt werden. Für Informationen über `IsSecureCommunication` siehe Unterstützung der sicheren S7-Kommunikation mit TLS (Seite 492)

Sie können beispielsweise folgenden Code verwenden, um ein Adressobjekt zu erstellen:

```
...
ConnectionConfiguration configuration = fingerprintDataProvider.Configuration;
ConfigurationMode configurationMode = configuration.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = configurationMode.PcInterfaces.Find("Intel (R)
Ethernet Connection I217-LM", 1);
// Create an address. This "ConfigurationAddress" is used as parameter for getting the
fingerprintData
ConfigurationAddress fingerprintAddress = pcInterface.Addresses.Create("192.68.0.1");
...
```

Die Konfiguration enthält eine Liste unterstützter Modes. Einer der Modes muss zum Abrufen von Fingerabdrücken ausgewählt werden. Der ausgewählte `ConfigurationMode` enthält eine Liste aller lokalen `PcInterfaces`, die den ausgewählten Mode unterstützen. Sie müssen eine der Schnittstellen auswählen. Die gewünschte Adresse kann in der Adressensammlung der ausgewählten `ConfigurationPcInterface` erstellt werden.

Programmcode: FingerprintData abrufen

Sie können die `FingerprintData` der Station abrufen, indem Sie die Aktion `GetFingerprintData` aufrufen. Die folgenden Parameter werden unterstützt:

Parameter	Beschreibung
<code>ConfigurationAddress</code>	Die Adresse des Geräts
<code>OnlineConfigurationDelegate</code>	Der Rückruf zur Behandlung von Inhibits

```
...
OnlineConfigurationDelegate preConfigurationDelegate = PreConfigureFingerprint;
FingerprintDataResult result =
fingerprintDataProvider.GetFingerprintData(fingerprintAddress , preConfigurationDelegate);
// The fingerprints
FingerprintDataItemComposition allFingerprints = result.FingerprintDataItems;
foreach (FingerprintDataItem item in allFingerprints)
{
// do something with the fingerprint ...
}
internal void PreConfigureFingerprint(OnlineConfiguration onlineConfiguration)
{
...
}
```

Delegate für die Online-Konfiguration

Die möglichen Typen von Online-Konfigurationen werden nachstehend aufgeführt:

Konfigurationsname	Beschreibung und Eigenschaften
OnlineConfiguration	- Basisklasse für andere Konfigurationen - OnlineConfiguration.Message: Zeichenkette (schreibgeschützte Eigenschaft, die die Konfigurationsmeldung enthält)

Da dies die Basisklasse für alle anderen Konfigurationen ist, ist dieser Eigenschaftsadapter deshalb in allen Konfigurationen verfügbar.

Der Datentyp der Konfiguration wird nachstehend aufgeführt:

Konfiguration	DataType	Beschreibung und Aktion
OnlineConfiguration	OnlineReadAccessPassword	Set password über Methode SetPassword(password:SecureString). Passwort eingeben, um Lesezugriff auf das Modul zu erhalten.

Eine nicht behandelte Konfiguration, die den Zugriff auf den Fingerabdruck verhindern kann, verursacht eine EngineeringDelegateInvocationException und bricht die Aktion ab. Bei einer nicht behandelten Ausnahme im Delegate wird eine EngineeringDelegateInvocationException ausgelöst.

Beispiel für die Implementierung von PreConfigureFingerprint

```
private static void PreConfigureFingerprint(OnlineConfiguration onlineConfiguration)
{
    OnlineReadAccessPassword readAccess = onlineConfiguration as OnlineReadAccessPassword;
    if (readAccess != null)
    {
        string passWD = "passWD";
        var password = new SecureString();
        foreach (var c in passWD) password.AppendChar(c);
        readAccess.SetPassword(password);
        return;
    }
    throw new NotSupportedException(); // Exception thrown in the delagate will cancel get fingerprints
}
```

FingerprintDataResult

Das von der Aktion GetFingerprintData zurückgegebene FingerprintDataResult liefert eine Zusammenstellung sämtlicher Fingerabdrücke für das Gerät.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Sie können beispielsweise folgenden Code verwenden, um Fingerabdruckelemente zu untersuchen:

```
{
...
FingerprintDataResult result = dataProvider.GetFingerprintData(fingerprintAddress,
preConfigurationDelegate);
FingerprintDataItemComposition allFingerprints = result.FingerprintDataItems;
foreach (FingerprintDataItem item in allFingerprints)
{
string fingerprintDataIdentifier = item.FingerprintDataIdentifier;
string fingerprintDataValue = plcItem.FingerprintDataValue;
// Do something, e.g. store Identifier and Value in a file
}
}
```

Fehlerbehandlung innerhalb der Openness-Fingerabdrücke

Alle Fehler innerhalb der Openness-Fingerabdrücke werden als wiederherstellbare Anwenderausnahmen `EngineeringTargetInvocationException` zugeordnet. Deshalb führt kein Fehler in einer Openness-Aktion zu einem Absturz des TIA Portals.

Voraussetzung ist, dass der Client die Ausnahme verarbeitet. Das kann beispielsweise wie in folgendem Code geschehen:

```
TiaPortal currentTIA = ...;
FingerprintDataProvider dataProviderFingerprints =
currentTIA.GetService<FingerprintDataProvider>();
OnlineConfigurationDelegate configurationDelegate = OnlineConfigurationFingerprintData;
try
{
// The used addr refers a PLC300.
FingerprintDataResult dataResult = dataProviderFingerprints.GetFingerprintData(addrObj,
configurationDelegate);
// Try to get a list of all FingerprintDataItems. But a PLC300 supports no fingerprints.
FingerprintDataItemComposition dataItems = dataResult.FingerprintDataItems;
foreach (FingerprintDataItem searchItem in dataItems)
{
string valueToCompareIdent = searchItem.FingerprintDataIdentifier;
string valueToCompareValue = searchItem.FingerprintDataValue;
DoAnything(valueToCompareIdent, valueToCompareValue);
}
}
catch (EngineeringTargetInvocationException targetException)
{
// In case of the not supported PLC this exception will be caught.
Console.WriteLine(targetException.Message);
}
```

Helper-Klassen

Für dieses Beispiel sind zwei Klassen nötig. Sie können folgenden Programmcode verwenden, um Fingerabdruck-Datenpaare zu kapseln.

```
/// <summary>
/// Helperclass to handle a "Dictionary" as serializeable class object
/// </summary>
[Serializable]
public class FingerprintDataPairs : Dictionary<string, string>, ISerializable
{
    public FingerprintDataPairs(){}
    public FingerprintDataPairs(SerializationInfo si, StreamingContext context)
    {
        KeyValuePair<string, string>[] dataPair = si.GetValue("KeyValuePairs",
            typeof(KeyValuePair<string, string>[])) as KeyValuePair<string, string>[];
        if (dataPair != null)
            foreach (KeyValuePair<string, string> pair in dataPair)
                this.Add(pair.Key, pair.Value);
    }
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Sie können folgendes Beispiel verwenden, um die Fingerabdruckdaten zu verarbeiten und einen Vergleich durchzuführen:


```

/// <summary>
/// Helperclass for handle the fingerprint data and compare two datasets.
/// </summary> public class FingerprintDataCompare
{
    internal FingerprintDataPairs m_FingerprintDataPairs;
    internal FingerprintDataPairs FingerprintDataPairSet
    {
        get
        {
            if (m_FingerprintDataPairs == null) m_FingerprintDataPairs = new FingerprintDataPairs();
            return m_FingerprintDataPairs;
        }
        set
        {
            m_FingerprintDataPairs = value;
        }
    }
    internal bool CompareDataSets(FingerprintDataPairs dataSet1, FingerprintDataPairs dataSet2)
    {
        foreach (string key in dataSet1.Keys)
        {
            if (dataSet2[key] != dataSet1[key])
                return false;
        }
        return true;
    }
    internal void SaveFingerprintDataItems(FingerprintDataPairs fingerprintDataPairSet, string
    filename)
    {
        using (FileStream fs = new FileStream(filename, FileMode.Create))
        {
            BinaryFormatter formatter = new BinaryFormatter();
            try
            {
                formatter.Serialize(fs, fingerprintDataPairSet);
            }
            catch (SerializationException e)
            {
                Console.WriteLine("Failed to write fingerprints. See: " + e.Message);
            }
            finally
            {
                fs.Close();
            }
        }
    }
    internal FingerprintDataPairs LoadFingerprintDataItems(string filename)
    {
        FingerprintDataPairs dataPair = null;
        using (FileStream fs = new FileStream(filename, FileMode.Open))
        {
            BinaryFormatter formatter = new BinaryFormatter();
            try
            {
                dataPair = (FingerprintDataPairs)formatter.Deserialize(fs);
            }
            catch (Exception e)
            {
                Console.WriteLine("Failed to load fingerprints. See: " + e.Message);
            }
        }
        return dataPair;
    }
}

```

```
}  
catch (System.Runtime.Serialization.SerializationException e)  
{  
    Console.WriteLine("Failed to read fingerprints. See: " + e.Message);  
}  
finally  
{  
    fs.Close();  
}  
}  
return dataPair;  
}  
}
```

Fingerabdrücke vergleichen

Um einen Vergleich durchzuführen, können Sie das `FingerprintDataResult` verwenden.

```
...  
// See chapter "FingerprintDataResult"  
FingerprintDataResult fingerprintDataResult = dataProvider.GetFingerprintData(address,  
preConfigurationDelegate);  
...
```

Um die Fingerabdrücke zu speichern, können Sie die Helper-Klassen verwenden:

```
string pathDat = "FingerprintDataOverview.dat";  
// Create a fingerprint comparer and fill the dataitems with the result of  
"GetFingerprintData".  
FingerprintDataCompare fingerprintDataSet = new FingerprintDataCompare();  
foreach (FingerprintDataItem item in fingerprintDataResult.FingerprintDataItems)  
    fingerprintDataSet.FingerprintDataPairSet.Add(item.FingerprintDataIdentifier,  
item.FingerprintDataValue);  
fingerprintDataSet.SaveFingerprintDataItems(fingerprintDataSet.FingerprintDataPairSet,  
pathDat);
```

Mit gespeicherten FingerprintDataItems und einem aktuellen FingerprintDataResult kann ein Vergleich der Fingerabdrücke durchgeführt werden:

```
...
string pathDat = "FingerprintDataOverview.dat";
FingerprintDataCompare fingerprintDataSet = new FingerprintDataCompare();
foreach (FingerprintDataItem item in fingerprintDataResult.FingerprintDataItems)
    fingerprintDataSet.FingerprintDataPairSet.Add(item.FingerprintDataIdentifier,
    item.FingerprintDataValue);
FingerprintDataPairs loadDataSet = new FingerprintDataPairs();
loadDataSet = fingerprintDataSet.LoadFingerprintDataItems(pathDat);
if (!fingerprintDataSet.CompareDataSets(loadDataSet,
    fingerprintDataSet.FingerprintDataPairSet))
{
    // Here is place to implement user actions.
}
```

5.11.2.7 Unter STEP 7 auf Cross Reference Service zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
-

Anwendung

Sie können mit TIA Portal Openness über die Methode GetService bei STEP 7-Objekten auf den CrossReferenceService zugreifen. Der Cross Reference Service stellt den IEngineeringService bereit, d. h. den CrossReferenceService bei anwendbaren STEP 7-Objekten. Wird bei nicht anwendbaren STEP 7-Objekten versucht, über die Methode GetService auf den CrossReferenceService zuzugreifen, gibt TIA Portal Openness einen Nullwert zurück.

Die folgenden STEP 7-Objekte unterstützen Cross Reference Openness:

- OB
- FB
- FC
- DB
- Instanz-DB
- Globaler DB
- Array-DB
- PLC-Variable

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

- PLC-Systemkonstante
- PLC-Anwenderdatentyp

Programmcode

```
Software softTarget = container.Software; //Accessing lower levels until we reach the
PlcBlockComposition
PlcSoftware plcSoft = (PlcSoftware)softTarget;
PlcBlockSystemGroup blockComposition = plcSoft.BlockGroup;
PlcBlockComposition blockFolder = blockComposition.Blocks;
PlcBlock mainBlock = blockComposition.Blocks.Find("Main");
if (mainBlock != null)
{
    try
    {
        CrossReferenceService crossReferenceService =
        mainBlock.GetService<CrossReferenceService>();
        .....
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

5.11.2.8 Querverweise für STEP 7 abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Querverweisinformationen zu anwendbaren STEP 7-Objekten bereitstellen. Die Openness API gibt das CrossReferenceResult mit den Textwerten der Benutzeroberfläche 'Querverweise' und das tatsächliche IEngineeringObject der anwendbaren STEP 7-Objekte zurück.

'Querverweise' versucht, das STEP 7-Objekt selbst bereitzustellen, für das Sie, unterstützt durch die jeweiligen TIA Portal Openness-Objekte, zusätzliche Aktionen ausführen können.

Hinweis

Derzeit wird der Typ `IEngineeringObject` von Mitgliedern des STEP 7-Objekts nicht unterstützt (Mitglieder des STEP 7-Objekts – DB-Mitglieder, FB-Mitglieder, FC-Mitglieder, Variablenmitglieder, UDT-Mitglieder); das `CrossReferenceResult` enthält jedoch sämtliche bereits verfügbaren Textinformationen.

Eigenschaften

Die folgende Tabelle enthält eine Liste mit von STEP 7- und WinCC Unified-Objekten in TIA Portal Openness unterstützten Eigenschaften:

Benutzerdefinierter Typ	Name der Eigenschaft	Datentyp	Lesen-/Schreibzugriff
SourceObject	Name	String	Lesen
	Address	String	Lesen
	TypeName	String	Lesen
	Device	String	Lesen
	Path	String	Lesen
	UnderlyingObject	Object	Lesen
ReferenceObject	Name	String	Lesen
	Address	String	Lesen
	TypeName	String	Lesen
	Device	String	Lesen
	Path	String	Lesen
	UnderlyingObject	Object	Lesen
Location	Name	String	Lesen
	ReferenceLocation	String	Lesen
	ReferenceType	Enum	Lesen
	Access	Enum	Lesen
	ReferencedAs	Object	Lesen
	ReferncedAsName	String	Lesen
	Address	String	Lesen
	TypeName	String	Lesen

Die folgende Tabelle enthält eine Liste mit von STEP 7- und WinCC Unified-Objekten in TIA Portal Openness unterstützten Navigatoren:

Benutzerdefinierter Typ	Navigator	Datentyp
CrossReferenceResult	Sources	SourceObjectComposition
SourceObject	References	ReferenceObjectComposition
ReferenceObject	Locations	LocationComposition

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Die folgende Tabelle enthält eine Liste mit von STEP 7- und WinCC Unified-Objekten in TIA Portal Openness unterstützten Methoden und Enumerationen:

Name der Funktion mit Signatur	Enum
public CrossReferenceResult GetCrossReferences(CrossReferenceFilter filter)	CrossReferenceFilter
	ReferenceType
	Access

Die folgende Tabelle enthält eine Liste mit von STEP 7- und WinCC Unified-Objekten in TIA Portal Openness unterstützten Filter-Enumerationen:

Enum	Werte
AllObjects	Alle Objekte anzeigen
ObjectsWithReferences	Objekt mit Verweisen anzeigen
ObjectsWithoutReferences	Objekt ohne Verweise anzeigen
UnusedObjects	Nicht verwendete Objekte anzeigen

Programmcode

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

```

Public void GetCrossReferences()
{
    CrossReferenceService crossReferenceService = MainBlock.GetService<CrossReferenceService>();
    CrossReferenceResult rootResultObject =
    xRefService.GetCrossReferences(CrossReferenceFilter.AllObjects);
    SourceObjectComposition rootResultSourceObjects = rootResultObject.Sources; //The API
    returns a single source object but the type is a composition keeping LTS in mind.
    PrintSourceObjects(rootResultSourceObjects);
}
Private void PrintSourceObjects(SourceObjectComposition sourceObjects)
{
    foreach(SourceObject source in sourceObjects)
    {
        // In this approach, cross reference tries to provide the actual client openness object via
        the "UnderlyingObject"attribute.
        // However, due to the fact that cross reference result can have an object of any type such
        as OB, FB, DB, Tag etc returned from the CrossReferenceService
        // and a limitation in openness that the type cannot be pre-mapped, the user must cast the
        object before accessing its attributes.
        // As a result, the openness user must convert sourceObject.UnderlyingObject to required
        EOM type (such as OB, DB, FB, Tag etc)
        if (source.UnderlyingObject is DataBlock)
        {
            var db = source.UnderlyingObject as DataBlock;//Perform actions on db.
        }
        else if(source.UnderlyingObject is CodeBlock)
        {
            var cb = source.UnderlyingObject as CodeBlock;
            //Perform actions on cb.
        }
        // Every new type for which support is added, the openness user must add the casting logic
        as shown in the above if-else conditional statement.
        // Currently, the EOM support is NOT available for Member objects such as Datablock member,
        Tag member etc.
        // In such cases, the "source.UnderlyingObject" will be null.
        Console.WriteLine(source.Name);
        //No other attributes are available from the IEngineeringObject. All the remaining values
        must be given from cross reference separately via SourceObject EOM.
        Console.WriteLine(source.Address);
        Console.WriteLine(source.TypeName);
        Console.WriteLine(source.Device);
        Console.WriteLine(source.Path);
        //Get the source object references
        ReferenceObjectComposition referenceObjects = source.References;
        PrintReferences(referenceObjects);
        //Get the source object's children
        SourceObjectComposition sourceObjectChildren = source.Children;
        PrintSourceObjects(sourceObjectChildren);
    }
}
private void PrintReferences(ReferenceObjectComposition referenceObjects)
{
    foreach (ReferenceObject referenceObject in referenceObjects)
    {
        // Needs similar casting for each type as seen in source object example snippet
    }
}

```



```

if (referenceObject.UnderlyingObject is DataBlock)
{
    var db = referenceObject.UnderlyingObject as DataBlock;
    //Perform actions on db.
}
else if(referenceObject.UnderlyingObject is CodeBlock)
{
    var cb = referenceObject.UnderlyingObject as CodeBlock;
    //Perform actions on cb.
}
//No other attributes are available from the IEngineeringObject. All the remaining values
must be given from cross reference separately via ReferenceObject EOM.
Console.WriteLine(referenceObject.Name);
Console.WriteLine(referenceObject.Address);
Console.WriteLine(referenceObject.TypeName);
Console.WriteLine(referenceObject.Device);
Console.WriteLine(referenceObject.Path);
LocationComposition locations = referenceObject.Locations;
//Looping through ReferenceLocations
foreach (Location location in locations)
{
    Console.WriteLine(location.Name);
    Console.WriteLine(location.ReferenceLocation);
    Console.WriteLine(location.ReferenceType);
    Console.WriteLine(location.Access);
    if(location.ReferencedAs is Tag)
    var referencedAsObject = location.ReferencedAs as Tag;
}
// Needs similar casting for each type as seen in source object example snippet
Console.WriteLine(location.ReferncedAsName);
Console.WriteLine(location.Address);
Console.WriteLine(location.TypeName);
}
}
}

```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.2.9 PLC von R/H-System online setzen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Mit dem Dienst RHOOnlineProvider können Sie entweder den primären PLC oder den Backup-PLC eines R/H-Systems online setzen.

Programmcode: Auf Dienst RHOOnlineProvider von einem Gerät aus zugreifen

Ändern Sie folgenden Code, um auf RHOOnlineProvider zuzugreifen:

```
Device device = project.Devices.Find("S7-1500R/H-System_1");  
RHOOnlineProvider rhOnlineProvider = device.GetService<RHOOnlineProvider>();
```

Programmcode: Verbindungsparameter festlegen

Mit dem Objekt ConnectionConfiguration können Sie eine Verbindung zum Gerät herstellen. Der Zugriff ist möglich über die Eigenschaft Configuration von RHOOnlineProvider. Weitere Informationen zur Herstellung der Verbindung finden Sie unter Auf Parameter einer Online-Verbindung zugreifen (Seite 451)

Ändern Sie den folgenden Programmcode, um über den Namen eine Verbindungsart einzurichten und auf eine PC-Schnittstelle zuzugreifen. Für Informationen über IsSecureCommunication siehe Unterstützung der sicheren S7-Kommunikation TLS (Seite 492)

```
ConnectionConfiguration connectionConfiguration = rhOnlineProvider.Configuration;  
ConfigurationMode mode = connectionConfiguration.Modes.Find("PN/IE");  
ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("Broadcom NetXtreme Gigabit Ethernet", 1);  
ConfigurationTargetInterface targetConfiguration = pcInterface.TargetInterfaces.Find("1 X1");  
bool success = connectionConfiguration.ApplyConfiguration(targetConfiguration);
```

Hinweis

R/H-System besteht aus zwei PLCs, eine einzelne Verbindungskonfiguration wird Ihnen bereitgestellt.

Programmcode: R/H-System online setzen

Sie können entweder den primären PLC oder den Backup-PLC online setzen. Wenn versucht wird, beide Ziele gleichzeitig online zu setzen, gibt das System eine Ausnahme EngineeringTargetException zurück.

Ändern Sie folgenden Programmcode, um den primären PLC online zu setzen:

```
OnlineState onlineState = rhOnlineProvider.GoOnlineToPrimary();
```

Ändern Sie folgenden Programmcode, um den Backup-PLC online zu setzen:

```
OnlineState onlineState = rhOnlineProvider.GoOnlineToBackup();
```

Hinweis

Wenn Sie einen PLC eines R/H-Systems online setzen, dürfen Sie ein zuvor gespeichertes Passwort wiederverwenden.

Programmcode: Onlinestatus eines R/H-Systems ermitteln

Sie können mit den Eigenschaften PrimaryState und BackupState von RHOnlineProvider den Online-Verbindungsstatus eines primären PLC und eines Backup-PLC getrennt voneinander feststellen. Beide Eigenschaften geben die Enumeration OnlineState zurück. Weitere Informationen dazu, wie Sie den Onlinezustand eines PLC ermitteln, finden Sie unter Zustand eines PLC feststellen (Seite 450)

Ändern Sie folgenden Programmcode, um den Zustand eines primären PLC und eines Backup-PLC zu ermitteln:

```
RHOnlineProvider rhOnlineProvider = ...;  
OnlineState primaryState = rhOnlineProvider.PrimaryState;  
OnlineState backupState = rhOnlineProvider.BackupState;
```

Programmcode: R/H-System offline setzen

Ändern Sie folgenden Programmcode, um ein R/H-System, das derzeit online ist, durch Aufruf der Methode RHOnlineProvider.GoOffline in den Offline-Zustand zu versetzen:

```
rhOnlineProvider.GoOffline();
```

Siehe auch

Auf Parameter einer Online-Verbindung zugreifen (Seite 451)

Status einer PLC ermitteln (Seite 450)

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Unterstützung der sicheren S7-Kommunikation mit TLS (Seite 492)

5.11.2.10 Auf Softwarebehälter über primären PLC eines R/H-Systems zugreifen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Für den Zugriff auf einen Softwarebehälter können Sie den primären PLC eines R/H-Systems verwenden. Beispielsweise stellt das R/H-System Softwarebehälter für einen primären PLC zur Verfügung, der als PLC_1 dargestellt wird. Wenn Sie jedoch versuchen, auf einen Softwarebehälter für einen Backup-PLC zuzugreifen, der als PLC_2 dargestellt wird, gibt das System null zurück.

Die Besonderheiten eines Softwarebehälters und seine Software-Eigenschaften werden unter Auf Software-Ziel zugreifen (Seite 150) beschrieben.

Programmcode: Auf Softwarebehälter zugreifen

Um auf Softwarebehälter über das primäre Gerät eines R/H-Systems zuzugreifen, ändern Sie den folgenden Programmcode:

```
foreach (DeviceItem deviceItem in rhDevice.DeviceItems)
{
    if (deviceItem.Name == "PLC_1")
    {
        SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
        ... //Work with softwareContainer
    }
}
```

Siehe auch

Auf Software-Ziel zugreifen (Seite 150)
Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

5.11.2.11 PLCs eines R/H-Systems laden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können mit der TIA Portal Openness-Anwendung sowohl in den primären PLC als auch in den Backup-PLC eines R/H-Systems laden. Sie können sowohl Hardware- als auch Softwarekomponenten des Systems laden. (Weitere Informationen finden Sie unter Hardware- und Softwarekomponenten ins PLC-Gerät laden (Seite 400))

Programmcode: RHDownloadProvider abrufen

Über RHDownloadProvider service können Sie aus einem Gerät in ein R/H-System laden.

Ändern Sie folgenden Programmcode, um RHDownloadProvider abzurufen:

```
...  
Device device = project.Devices.Find("S7-1500R/H-System_1");  
RHDownloadProvider rhDownloadProvider = device.GetService<RHDownloadProvider>();  
...
```

Hinweis

Auf den Dienst DownloadProvider wird nicht im Zusammenhang mit CPUs zugegriffen, die zum R/H-System gehören.

Programmcode: IConfiguration abrufen

RHDownloadProvider stellt das Objekt ConnectionConfiguration über die Eigenschaft Configuration zur Verfügung, mit deren Hilfe die Verbindung zum Gerät konfiguriert wird. Für Informationen über IsSecureCommunication siehe Unterstützung der sicheren S7-Kommunikation TLS (Seite 492)

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie folgenden Programmcode, um das Objekt IConfiguration über ConnectionConfiguration an RHDownloadProvider abzurufen:

```
...
RHDownloadProvider rhDownloadProvider = device.GetService<RHDownloadProvider>();
ConnectionConfiguration connectionConfiguration = rhDownloadProvider.Configuration;
ConfigurationMode mode = connectionConfiguration.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("Broadcom NetXtreme Gigabit Ethernet", 1);
ICConfiguration targetConfiguration = pcInterface.TargetInterfaces.Find("1 X1");
...
```

Hinweis

R/H Systeme bestehen aus zwei PLCs. Es wird nur ein Verbindungskonfigurationsobjekt bereitgestellt, das zum Laden sowohl in primäre PLCs als auch in Backup-PLCs verwendet werden kann.

Programmcode: In primäre CPU und Backup-CPU laden

Ändern Sie folgenden Programmcode, um durch Aufrufen von RHDownloadProvider.DownloadToPrimary in den primären PLC zu laden:

```
DownloadResult DownloadToPrimary(configuration, preDownloadConfigurationDelegate,
postDownloadConfigurationDelegate, downloadOptions);
```

Ändern Sie folgenden Programmcode, um durch Aufrufen von RHDownloadProvider.DownloadToBackup in den Backup-PLC zu laden:

```
DownloadResult DownloadToBackup(configuration, preDownloadConfigurationDelegate,
postDownloadConfigurationDelegate, downloadOptions);
```

Parameter der Methode RHDownloadProvider

Sowohl RHDownloadProvider.DownloadToPrimary als auch RHDownloadProvider.DownloadToBackup akzeptieren dieselben Parameter und geben ebenfalls ein DownloadResult zurück. Weitere Informationen zu den Details von IConfiguration, DownloadConfigurationDelegate, DownloadOptions und DownloadResult finden Sie unter Hardware- und Softwarekomponenten ins PLC-Gerät laden (Seite 400)

Parametername	Typ	Beschreibung
configuration	Siemens.Engineering.Connection.IConfiguration	Konfiguration der Verbindung mit einem Gerät.
onlineAddress	Siemens.Engineering.Connection.ConfigurationAddress	Optionaler Parameter für das Laden in eine geänderte IP-Adresse
preDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Delegate, der aufgerufen wird, um die Konfiguration vor dem Laden zu prüfen

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Parametername	Typ	Beschreibung
postDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Delegate, der aufgerufen wird, um die Konfiguration nach dem Laden zu prüfen
downloadOptions	Siemens.Engineering.Download.DownloadOptions	Download-Optionen

Sie haben je nach Zustand des R/H-Systems die Möglichkeit, über DownloadConfigurations für den Ladevorgang einen Stopp des Systems anzufordern. Deshalb werden zusätzlich zu der Konfiguration, die unter Hardware- und Softwarekomponenten ins PLC-Gerät laden (Seite 400) beschrieben ist, auch die folgenden Datentypen zur Unterstützung von RHDnload hinzugefügt.

Konfiguration	Datentyp	Aktion	Beschreibung
DownloadSelection-Configuration	StopHSystem	Set CurrentSelection:StopHSystemSelections. Verfügbare Enumerationswerte: - NoAction (Keine Aktion) - StopHSystem (R/H-System stoppen)	Die Module werden gestoppt, um das Laden in ein Gerät zu ermöglichen.
	StopHSystemOrModule	Set CurrentSelection:StopHSystemOrModuleSelections. Verfügbare Enumerationswerte: - NoAction (Keine Aktion) - StopHSystem (R/H-System stoppen) - StopModule (Modul stoppen)	Die Module werden gestoppt, um das Laden in ein Gerät zu ermöglichen.
	StartBackupModules	Set CurrentSelection:StartBackupModulesSelections. Verfügbare Enumerationswerte: - NoAction (Keine Aktion) - SwitchToPrimaryCpu (Auf primär wechseln) - StartModule (Modul starten)	Module nach dem Laden ins Gerät starten.
	SwitchBackupToPrimary	Set CurrentSelection:SwitchBackupToPrimarySelections. Verfügbare Enumerationswerte: - NoAction (Keine Aktion) - SwitchToPrimaryCpu (Auf primär wechseln)	Module nach dem Laden ins Gerät starten.

Programmcode: Ladekonfiguration in Rückrufen handhaben

Ändern Sie folgenden Programmcode in Aufrufe von DownloadToPrimary und DownloadToBackup während der Behandlung von Konfigurationen in Rückrufen:

Beispiel für den Aufruf von Download

```

static void Main(string[] args)
{
    ...
    Project project = tiaPortal.Projects[0];
    Device device = project.Devices.Find("S7-1500R/H-System_1");
    RHDownloadProvider rhDownloadProvider = device.GetService<RHDownloadProvider>();
    ConnectionConfiguration connectionConfiguration = rhDownloadProvider.Configuration;
    ConfigurationMode mode = connectionConfiguration.Modes.Find("PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("Broadcom NetXtreme Gigabit Ethernet", 1);
    IConfiguration targetConfiguration = pcInterface.TargetInterfaces.Find("1 X1");
    // Download to primary
    DownloadResult primaryDownloadResult =
    rhDownloadProvider.DownloadToPrimary(targetConfiguration,
    PreConfigureDownloadCallback,
    PostConfigureDownloadCallback,
    DownloadOptions.Hardware | DownloadOptions.Software);
    WriteDownloadResults(primaryDownloadResult);
    // Download to backup
    DownloadResult backupDownloadResult =
    rhDownloadProvider.DownloadToBackup(targetConfiguration,
    PreConfigureDownloadCallback,
    PostConfigureDownloadCallback,
    DownloadOptions.Hardware | DownloadOptions.Software);
    WriteDownloadResults(backupDownloadResult);
    ...
}

private static void PreConfigureDownloadCallback(DownloadConfiguration
downloadConfiguration)
{
    StopHSystem stopHSystem = downloadConfiguration as StopHSystem;
    if (stopHSystem != null)
    {
        stopHSystem.CurrentSelection = StopHSystemSelections.StopHSystem;
    }
    OverwriteTargetLanguages overwriteTargetLanguages = downloadConfiguration as
    OverwriteTargetLanguages; if (overwriteTargetLanguages != null)
    {
        overwriteTargetLanguages.Checked = true;
    }
    AlarmTextLibrariesDownload alarmTextLibraries = downloadConfiguration as
    AlarmTextLibrariesDownload;
    if (alarmTextLibraries != null)
    {
        alarmTextLibraries.CurrentSelection =
        AlarmTextLibrariesDownloadSelections.ConsistentDownload;
        return;
    }
    CheckBeforeDownload checkBeforeDownload = downloadConfiguration as CheckBeforeDownload;
    if (checkBeforeDownload != null)
    {
        checkBeforeDownload.Checked = true;
        return;
    }
}

```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Beispiel für den Aufruf von Download

```
ConsistentBlocksDownload consistentBlocksDownload = downloadConfiguration as
ConsistentBlocksDownload;
if (consistentBlocksDownload != null)
{
    consistentBlocksDownload.CurrentSelection =
    ConsistentBlocksDownloadSelections.ConsistentDownload;
    return;
}
OverwriteSystemData overwriteSystemData = downloadConfiguration as OverwriteSystemData;
if (overwriteSystemData != null)
{
    overwriteSystemData.CurrentSelection = OverwriteSystemDataSelections.Overwrite;
    return;
}
}
private static void PostConfigureDownloadCallback(DownloadConfiguration
downloadConfiguration)
{
    StartModules startModules = downloadConfiguration as StartModules;
    if (startModules != null)
    {
        startModules.CurrentSelection = StartModulesSelections.StartModule;
        return;
    }
}
private static void WriteDownloadResults(DownloadResult result)
{
    Console.WriteLine("State:" + result.State);
    Console.WriteLine("Warning Count:" + result.WarningCount);
    Console.WriteLine("Error Count:" + result.ErrorCount);
    RecursivelyWriteMessages(result.Messages);
}
private static void RecursivelyWriteMessages(DownloadResultMessageComposition messages,
string indent = "")
{
    indent += "\t";
    foreach (DownloadResultMessage message in messages)
    {
        Console.WriteLine(indent + "DateTime: " + message.DateTime);
        Console.WriteLine(indent + "State: " + message.State);
        Console.WriteLine(indent + "Message: " + message.Message);
        RecursivelyWriteMessages(message.Messages, indent);
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Herunterladen zu den PLCs (Seite 400)

Unterstützung der sicheren S7-Kommunikation mit TLS (Seite 492)

5.11.2.12 Online-Verbindung zur PLC aufbauen oder trennen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Alle Geräte werden enumeriert.
Siehe Geräteelemente aufrufen (Seite 375).

Verwendung

Sie können die Online-Verbindung zu einer PLC herstellen oder eine bestehende Online-Verbindung trennen.

Programmcode

Um die Online-Verbindung zu einer PLC herzustellen oder zu trennen, ändern Sie folgenden Programmcode:

```
public static void SetOnlineConnection(DeviceItem deviceItem)
{
    OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
    if (onlineProvider == null) { return; }
    // Go online
    if (onlineProvider.Configuration.IsConfigured)
    {
        onlineProvider.GoOnline();
    }
    // Go offline
    onlineProvider.GoOffline();
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Sie können auch die Online-Verbindungen zu allen verfügbaren PLCs in einem Projekt herstellen oder trennen.

```
public static void SetOnlineConnectionForAllPLCs(Project project)
{
    foreach (Device device in project.Devices)
    {
        foreach (DeviceItem deviceItem in device.DeviceItems)
        {
            OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
            if (onlineProvider != null)
            {
                // Establish online connection to PLC:
                onlineProvider.GoOnline();

                // ...

                // Disconnect online connection to PLC:
                onlineProvider.GoOffline();
            }
        }
    }
}
```

5.11.2.13 Projektsprache der PLC zuordnen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Sie können mithilfe von TIA Portal Openness dem Webserver Projektsprachen zuordnen und Sprachen einer S7-1500 PLC anzeigen. Hierzu ist ein dynamisches Attribut vom Typ `StructuredData` notwendig.

Für den Zugriff auf die Einstellungen in mehreren Sprachen wurde das neue dynamische Attribut `MultilingualSupport` vom Typ `TableData` hinzugefügt. Dieses Attribut kann in Zeilen aufgeteilt werden. In den einzelnen Zeilen kann die zugeordnete Projektsprache jeweils mit dem dynamischen Attribut `ProjectLanguage` festgelegt oder gelesen werden.

Programmcode

Ändern Sie folgenden Programmcode, um die Projektsprache der PLC zuzuordnen:

```
//To Set the Project Language
LanguageSettings languageSettings = project.LanguageSettings;
LanguageAssociation activeLanguages = languageSettings.ActiveLanguages;
DeviceItem Plc = ...;
Tabledata multilingualSupportTable = Plc.GetAttribute("MultilingualSupport");
StructuredDataComposition structuredDataComp = multilingualSupportTable.Rows; // This will
return a collection of Structured Data.
StrcturedData firstRow= structuredDataComp[0]; //First row
Language activeGermanLanguage = activeLanguages.Find(CultureInfo.GetCultureInfo("de-DE"));
firstRow.SetAttribute("ProjectLanguage", activeGermanLanguage.Culture);
//To Get the Languages of device displayed
var assignedToGerman = firstRow.GetAttribute("DisplayLanguage");
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.2.14 Beobachtungs- und Forcetabellen für Webserver- und PLC-Display zuordnen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Mit TIA Portal Openness können Sie bereits erstellte Beobachtungs- und Forcetabellen dem Webserver zuordnen. Bei der Beobachtungstabelle und der Forcetabelle handelt es sich um Software, die im Softwarecontainer gefunden, erstellt und gelöscht werden kann. Sie können mit TIA Portal Openness Beobachtungs- und Forcetabellen exportieren/importieren. Zum Export/Import von Beobachtungs- und Forcetabellen siehe Beobachtungs- und Forcetabelle exportieren/importieren (Seite 1431)

Für die Zuordnung von Beobachtungs- und Forcetabellen zum Webserver wird der Dienst WatchAndForceTableAccessManager am DeviceItem der PLC verwendet.

Dieser Dienst enthält zwei Navigatoren für Beobachtungs- und Forcetabellen.

- WatchtableAccessRules {WatchtableAccessRuleComposition}
- ForcetableAccessRules {ForcetableAccessRuleComposition}

Der Navigator WatchTableAccessRules stellt die Zusammensetzung WatchTableAccessRuleComposition zur Verfügung, die Objekte vom Typ

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

WatchTableAccessRule enthält. Die Zusammensetzung ist standardmäßig leer. Für die WatchTableAccessRule -Objekte sind die Aktionen "Finden" und "Erstellen" definiert.

AccessRule für WatchTable & ForceTable

- **PlcWatchTable** watchtable {get;} gibt die Beobachtungstabelle des Softwareobjekts zurück. Um eine zugeordnete Beobachtungstabelle auszutauschen, muss der Benutzer die Zuordnung der aktuellen Tabelle entfernen (mit WatchTableAccessRule.Delete()) und eine neue Tabelle hinzufügen (mit WatchTableAccessRule.Create()).
- **WatchAndForceTableAccess** access {get; set;} gibt die Webserver-Zugriffsstufe (d. h. Lesen oder Lesen/Schreiben) zurück bzw. legt sie fest

Name in der TIA UI	Wert	enum-Beschreibung in Openness
-	0	None
Lesen	1	Read
Lesen/Schreiben	2	Write

In folgenden Fällen werden Fehler ausgelöst:

- Wenn der Webserver nicht aktiviert wurde (WebserverActivate == FALSE), wird eine EngineeringTargetInvocationException ausgelöst, deren Fehlerdetails besagen, dass eine WatchTableAccessRule nur hinzugefügt werden kann, wenn der Webserver aktiviert ist.
- Wenn der Benutzer versucht, eine Beobachtungstabelle mit WatchAndForceTableAccess.None hinzuzufügen, wird eine ConfigOpennessUserException ausgelöst, deren Fehlerdetails besagen, dass WatchAndForceTableAccess.None als Zugriffsstufe nicht zulässig ist

Programmcode: Beobachtungstabelle zuordnen und deren Zuordnung aufheben

Ändern Sie folgenden Programmcode, um eine Beobachtungstabelle auf dem Webserver zu suchen:

```
WatchAndForceTableAccessManager mngr =
deviceItem.GetService<WatchAndForceTableAccessManager>();
WatchTableAccessRuleComposition watchTableCmp = mngr.WatchTableAccessRules;
WatchTableAccessRule accessRule1 = watchTableCmp.Find(watchTable1);
```

Hinweis

Es wird null zurückgegeben, wenn die betreffende Beobachtungstabelle nicht mit einer WatchTableAccessRule auf dem Webserver verknüpft ist

Ändern Sie folgenden Programmcode, um auf dem Webserver eine neue Beobachtungstabelle mit Lesezugriff zu erstellen:

```
WatchAndForceTableAccessManager mngr =
deviceItem.GetService<WatchAndForceTableAccessManager>();
WatchTableAccessRuleComposition watchTableCmp = mngr.WatchTableAccessRules;
watchTableCmp.Create(watchTable1, WatchAndForceTableAccess.Read);
```

Ändern Sie folgenden Programmcode, um eine vorhandene WatchTableAccessRule zu löschen und die Zuordnung der Beobachtungstabelle zum Webserver aufzuheben:

```
WatchTableAccessRule whatchtable= watchTableCmp.Find(watchTable1);
whatchtable.Delete();
```

In folgenden Fällen werden Fehler ausgelöst:

- Wenn der Webserver nicht aktiviert wurde (WebserverActivate == FALSE), wird eine EngineeringTargetInvocationException ausgelöst, deren Fehlerdetails besagen, dass eine WatchTableAccessRule nur hinzugefügt werden kann, wenn der Webserver aktiviert ist
- Wenn eine WatchtableAccessRule zu dieser Beobachtungstabelle bereits vorhanden ist, wird eine ConfigOpenessUserException ausgelöst, deren Fehlerdetails besagen, dass die Beobachtungstabelle bereits vorhanden ist.
- Wenn Sie versuchen, eine Beobachtungstabelle mit WatchAndForceTableAccess.None hinzuzufügen, wird eine ConfigOpenessUserException ausgelöst, deren Fehlerdetails besagen, dass WatchAndForceTableAccess.None als Zugriffsstufe nicht zulässig ist

Programmcode: Dem Webserver eine Forcetabelle zuordnen

Ändern Sie folgenden Programmcode, um eine Forcetabelle auf dem Webserver zu suchen:

```
WatchAndForceTableAccessManager mngr =
deviceItem.GetService<WatchAndForceTableAccessManager>();
ForceTableAccessRuleComposition forceTableCmp = mngr.ForceTableAccessRules;
ForceTableAccess forceTable = forceTableCmp.Find(forceTable1);
```

Ändern Sie folgenden Programmcode, um auf dem Webserver eine PLC-Forcetabelle mit Lesezugriff zu erstellen:

```
ForceTableAccessRuleComposition forceTableCmp = mngr.ForceTableAccessRules;
forceTableCmp.Create(forceTable1, WatchAndForceTableAccess.Read);
```

Beobachtungs- und Forcetabellen am PLC-Display

Sie können denselben Openness-Dienst WatchAndForceTableAccessManager verwenden, der am Submodul Display (DeviceItem) des PLC DeviceItem verfügbar ist. Für die Zuordnung von Beobachtungs- und Forcetabellen am PLC-Display wird dieselbe Webserver-Funktion wie oben beschrieben verwendet. Im Unterschied zum Webserver kann das Display nicht deaktiviert werden. Damit ist eine ähnliche Validierung wie bei WebserverActive hier nicht möglich.

Siehe auch

Beobachtungs- und Forcetabelle exportieren/importieren (Seite 1431)

Verbindung zum TIA Portal aufbauen (Seite 84)

5.11.2.15 Zertifikat verwalten

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können mit TIA Portal Openness folgende Aufgaben mit Zertifikaten ausführen:

- Zertifikat erstellen und löschen
- Zertifikat exportieren und importieren
- Zertifikate zuordnen und ihre Zuordnung aufheben
- Zertifikat-ID abrufen

Local Certificate Manager

Für Gerätezertifikate wird der Dienst LocalCertificateManager an der PLC des DeviceItem bereitgestellt. Der LocalCertificateManager verfügt über einen lokalen Speicher namens LocalCertificateStore für die Zertifikate. Darin werden die Zertifikate für die bestimmte PLC gespeichert. Um das Geräteelement abzurufen, das Inhaber der Dienstinstanz ist, können Sie die Eigenschaft OwnedBy verwenden.

```
DeviceItem Plc= ...;
// Get local certificate manager
var localCertificateManager = Plc.GetService<LocalCertificateManager>();
// Disable global certificate manager
localCertificateManager.EnableGlobalCertificatesStore = false;
// Get local certificate store
var localCertificateStore = localCertificateManager.LocalCertificateStore;
```


Zertifikateverwaltung

Ändern Sie folgenden Programmcode, um ein Zertifikat zu erstellen:

```
/ Create templates
var templateTls = localCertificateStore.GetCertificateTemplate(CertificateUsage.Tls);
var templateWebserver =
localCertificateStore.GetCertificateTemplate(CertificateUsage.WebServer);
var templateOpcUaServer =
localCertificateStore.GetCertificateTemplate(CertificateUsage.OpcUaServer); //
...
//Template handling and configuration is handled later
// Create certificates
var certificateTls = localCertificateStore.Certificates.Create(templateTls);
var certificateWeb = localCertificateStore.Certificates.Create(templateWebserver);
var certificateOpcUa = localCertificateStore.Certificates.Create(templateOpcUaServer);
```

Ändern Sie folgenden Programmcode, um ein Zertifikat zu löschen und zu exportieren:

```
var exportPath = ...;
// renew a certificatecertificate
Tls.Delete();
certificateTls = localCertificateStore.Certificates.Create(templateTlsNew);
certificateWeb.Export(new FileInfo(exportPath));
```

Hinweis

Bei TIA Portal Openness V19 akzeptiert Export() jetzt exportPath als Parameter und ermöglicht den Export von Zertifikaten in einem beliebigen Format.

Ändern Sie folgenden Programmcode, um mit dynamischen Attributen dem Webserver und OPC UA-Server Zertifikate zuzuordnen:

```
DeviceItem opcUaSubmodule = ...;
//Find assigned certificates
var foundWebCertificate = Plc.GetAttribute("WebserverCertificate");
var foundOpcUaCertificate = opcUaSubmodule.GetAttribute("OpcUaServerCertificate");
//Assign certificatesPlc.SetAttribute("WebserverCertificate", certificateWeb);
opcUaSubmodule.SetAttribute("OpcUaServerCertificate", certificateOpcUa);
//Unassign certificatesPlc.SetAttribute("WebserverCertificate", null);
opcUaSubmodule.SetAttribute("OpcUaServerCertificate", null);
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie folgenden Programmcode, um Zertifikate in den lokalen Zertifikatsspeicher für eine PLC zu importieren:

```
var certificateWithoutPwd = ...;
var certificateWithPwd = ...;
var password = new SecureString();
//
...
// Import certificates
// Without password
var importedCertificate1 = certificates.Import(new FileInfo(certificateWithoutPwd));
//With password
var importedCertificate2 = certificates.Import(new FileInfo(certificateWithPwd), password);
```

Ändern Sie folgenden Programmcode, um die Zertifikats-ID abzurufen:

```
var certificateId = importedCertificate2.Id;
```

Ändern Sie folgenden Programmcode, um anzugeben, ob ein Zertifikat über einen privaten Schlüssel verfügt. Hierzu kann die boolesche Eigenschaft HasPrivateKey am Zertifikatsobjekt verwendet werden.

```
if(importedCertificate2.HasPrivateKey)
{
//Do something
}
```

Zertifikatsvorlage

Die Vorlagen dienen als Grundlage für die Erstellung von Zertifikaten. Neue Zertifikate können im LocalCertificateStore mit der Aktion GetCertificateTemplate(CertificateUsage) erstellt werden. Hierbei wird CertificateUsage mit den folgenden möglichen Werten enumeriert:

- None (nicht unterstützt)
- Tls
- WebServer
- OpcUaServer
- OpcUaClient
- OpcUaClientServer

```
// Create template
var certTemplate = localCertificateStore.GetCertificateTemplate(CertificateUsage.Tls);
```

In einer Zertifikatsvorlage können alle Eigenschaften eines Zertifikats festgelegt werden. Auf die Zertifikatsvorlage besteht Lese- und Schreibzugriff:

- Signatur: Der verwendete Signatur-Algorithmus gehört zum Typ `SignatureAlgorithm`, der mit den folgenden möglichen Werten enumeriert wird:
 - `None` (nicht unterstützt)
 - `Sha1RSA`
 - `Sha256RSA`

```
certTemplate.Signature = SignatureAlgorithm.Sha1RSA;
```

- `SubjectAlternativeNames`: Dies ist eine Zusammensetzung, die Objekte vom Typ `SubjectAlternativeName` (IList vom Typ `<SubjectAlternativeName>`) enthält. Ein neuer SAN kann mit der Aktion `Create(SubjectAlternativeNameType, Zeichenkette)` erstellt werden. Die übergebenen Parameter sind der Type (vom Typ `SubjectAlternativeNameType`) und der Value (vom Typ `String`).
- `SubjectAlternativeNameType`
 - `None` (nicht unterstützt)
 - `Dns`
 - `Email`
 - `IP`
 - `Uri`

```
var san1 = certTemplate.SubjectAlternativeNames.Create(SubjectAlternativeNameType.Dns,
"127.0.0.1");
var san2 = certTemplate.SubjectAlternativeNames.Create(SubjectAlternativeNameType.IP,
"192.168.178.1");
var san3 = certTemplate.SubjectAlternativeNames.Create(SubjectAlternativeNameType.Email,
"test@prov.com");
var san4 = certTemplate.SubjectAlternativeNames.Create(SubjectAlternativeNameType.Uri,
"something");
san3.Delete();
```

- `SubjectCommonName`: `string`

```
string subjectCommonName = certTemplate.SubjectCommonName;
certTemplate.SubjectCommonName = "exampleSubjectCommonName";
```

Verwendung des Typs `CertificateUsage`

```
var usage = certTemplate.Usage;
certTemplate.Usage = CertificateUsage.OpcUaClientServer;
```

`ValidFrom`: Typ `DateTime`

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

ValidUntil: Typ DateTime

```
var validFrom = certTemplate.ValidFrom;  
var validUntilDateTime = new DateTime(2080, 10, 10);  
certTemplate.ValidUntil = validUntilDateTime;
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.2.16 Webserver-Benutzer von SIWAREX-Modulen verwalten

Voraussetzung

- TIA Portal ist mit TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness für Webserver-Benutzer von SIWAREX-Modulen folgende Funktionen verwalten:

- Webserver-Benutzer aktivieren oder deaktivieren
- Namen des Benutzers ändern
- Berechtigungen des Benutzers festlegen
- Passwort des Benutzers festlegen

Mit der Funktion SimpleWebserverUserManagement können Sie die Liste der Webserver-Benutzer von SIWAREX-Modulen abrufen. Die Funktion wird für SIWAREX-Module mit einem Webserver-Benutzer unterstützt.

Unterstützte Module

Derzeit wird die Funktion nur von zwei Modulen der ET200SP-Familie verwendet:

- 7MH4 138-6CA00-OCU0 V1.0 – TM SIWAREX WP341 HF
- 7MH4 138-6BA00-OCU0 V1.0 – TM SIWAREX WP351 HF

Eigenschaft

Name der Eigenschaft	Datentyp	Beschreibung
UserName	String	Zum Festlegen oder Abrufen des Benutzernamens
Active	Boolean	Zum Hinzufügen der Aktivierung des Benutzers
Permissions	SimpleWebserverUserPermissions	Zum Festlegen oder Abrufen der Benutzerberechtigungen SimpleWebserverUserPermissions unterstützt die folgenden Enumerationen: - SimpleWebserverUserPermissions.ReadWrite - SimpleWebserverUserPermissions.ReadOnly - SimpleWebserverUserPermissions.None (bei deaktiviertem Benutzer) Eine EngineeringTargetInvocationException wird ausgelöst, wenn die falsche Berechtigung festgelegt wurde.
SetPassword()	Secure String	Zum Festlegen des Benutzerpassworts. Das Passwort kann nur bei einem aktiven Benutzer festgelegt werden.

Programmcode

```
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.SimpleWebserverUser;
private void SettingWebserverUser()
{
    //To get the list of webserver users
    SimpleWebserverUserManagement feature =
    subModule.GetService<SimpleWebserverUserManagement>();
    // To get the specific user from the list of webserver users
    SimpleWebserverUser webserverUser = feature.WebserverUsers[1];
    // To set activation of the user
    webserverUser.active = true;
    // To set the username of the user
    webserverUser.UserName = "userName1";
    // To get the username of the user
    string name = webserverUser.UserName;
    // To set the permissions to Read for the user
    webserverUser.Permissions = SimpleWebserverUserPermissions.ReadOnly;
    SimpleWebserverUserPermissions permission = webserverUser.Permissions;
    // To set the password of the user
    webserverUser.SetPassword(GetSecureString("Admin123"));
}
```

5.11.2.17 Verwalten der Syslog-Konfiguration

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit dem Dienst SysLogConfigurationManager in TIA Portal Openness können Sie die SysLog-Funktion aktivieren und deaktivieren. Diese Funktion ermöglicht es Ihnen, verschiedene Transportprotokolle auszuwählen und verschiedene Server für SysLog zu konfigurieren.

Die Funktion ist ab der S7-1500 PLC Firmware-Version 3.1 erhältlich.

Eigenschaft

Der Dienst SysLogConfigurationManager bietet die folgenden Eigenschaften zur Änderung der SysLog-Konfiguration.

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
EnableSystemLogging	Boolean	Aktivieren oder Deaktivieren der SysLog-Einstellung der SysLog-Konfiguration	Lesen/Schreiben
TransportProtocol	SysLogTransportProtocolType	Gibt die Konstanten für die Optionen des Transportprotokolls in der SysLog-Konfiguration an. Die folgenden Enum-Elemente und ihre entsprechenden Werte werden unterstützt: • Keiner - 0 • TLSServerAndClientAuthentication - 1 • TLSOnlyServerAuthentication - 2 • UDP - 3	Lesen / Schreiben
SysLogServerConfiguration	SysLogServerConfigurationComposition	Empfängt oder gibt alle Servereinzelheiten der SysLog-Konfiguration an. Einzelheiten zur Eigenschaft SysLogServerConfiguration finden Sie im folgenden Abschnitt SysLogServerConfiguration	Lesen
SysLogAutoAcceptClient	Boolean	Aktivieren oder Deaktivieren des automatischen Akzeptanzzertifikats in der Runtime-Einstellung der SysLog-Konfiguration	Lesen/Schreiben

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
SysLogClientCertificateld	UInt	Stellt die Zertifikat-ID des Clients für die SysLog-Konfiguration dar	Lesen/Schreiben
SysLogTrustedCertificatelds	String	Stellt die vertrauenswürdige Zertifikat-ID für die SysLog-Konfiguration dar	Lesen/Schreiben

Eigenschaft von SysLogServerConfiguration

SysLogServerConfiguration stellt die Serverkonfiguration der SysLog-Konfigurationen dar.

- SysLogServerConfiguration Find(string sysLogServerAddress)
- SysLogServerConfiguration Create(string sysLogServerAddress, UInt16 sysLogServerPort)

Name der Eigenschaft	Datentyp	Beschreibung
SysLogServerAddress	String	Stellt die Serveradresse der SysLog-Konfiguration dar
SysLogServerPort	UInt16	Stellt die Portnummer der SysLog-Konfiguration dar

Programmcode

Um die SysLog-Einstellung der SysLog-Konfiguration zu aktivieren oder zu deaktivieren, ändern Sie folgenden Programmcode:

```
// To access EnableSysLog Property
var service = plc.GetService<SysLogConfigurationManager>();
service.EnableSystemLogging = true;
```

Um das Transportprotokoll der SysLog-Konfiguration abzurufen oder festzulegen, ändern Sie folgenden Programmcode:

```
var service = plc.GetService<SysLogConfigurationManager>();
service.EnableSystemLogging = true;
service.TransportProtocol = SysLogTransportProtocolType.TLSServerAndClientAuthentication;
```

Um alle Servereinzelheiten der SysLog-Konfiguration abzurufen oder festzulegen, ändern Sie folgenden Programmcode:

```
var service = plc.GetService<SysLogConfigurationManager>();
service.EnableSystemLogging = true;
service.SysLogServerConfiguration.Create("www.test.com", 100);
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Um die Zertifikatakzeptanz in der Runtime-Einstellung zu aktivieren oder zu deaktivieren, ändern Sie folgenden Programmcode:

```
var service = plc.GetService<SysLogConfigurationManager>();
service.TransportProtocol = SysLogTransportProtocolType.TLSOnlyServerAuthentication;
plc.SetAttribute("SysLogAutoAcceptClient", true);
```

Um die vertrauenswürdige Zertifikat-ID für die SysLog-Konfiguration darzustellen, ändern Sie folgenden Programmcode:

```
var service = plc.GetService<SysLogConfigurationManager>();
service.EnableSystemLogging = true;
service.TransportProtocol = SysLogTransportProtocolType.TLSServerAndClientAuthentication;
plc.SetAttribute("SysLogTrustedCertificateIds", certificate.Id.ToString());
```

Um die Zertifikat-ID des Clients für die SysLog-Konfiguration darzustellen, ändern Sie folgenden Programmcode:

```
var service = plc.GetService<SysLogConfigurationManager>();
service.EnableSystemLogging = true;
service.TransportProtocol = SysLogTransportProtocolType.TLSServerAndClientAuthentication;
plc.SetAttribute("SysLogClientCertificateId", (uint)1);
```

5.11.2.18 Unterstützung der sicheren S7-Kommunikation mit TLS

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können TIA Portal Openness für das Einrichten der Parameter für eine Online-Verbindung verwenden.

Diese Online-Verbindungen werden im Objekt ConnectionConfiguration eingerichtet, auf das Sie über den Konfigurationsnavigator des OnlineProvider-Dienstes zugreifen können. Der Dienst ist auf einem DeviceItem vorhanden, der eine PLC darstellt.

```
DeviceItem deviceItem = ...;
OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
ConnectionConfiguration configuration = onlineProvider.Configuration;
```

Abhängig von den verfügbaren Verbindungen für eine PLC können Sie durch die verfügbaren Verbindungsarten enumerieren oder eine bestimmte Verbindungsart festlegen. Für jede Verbindungsart können Sie dann die PC-Schnittstelle für diesen Verbindungstyp festlegen.

Die Aktion Find in PcInterfaces hat zwei Parameter. Zuerst den Namen der Schnittstellenbaugruppe (ConfigurationPcInterface) und darüber hinaus die Nummer der Schnittstellenbaugruppe. Die Schnittstellenbaugruppe hat typischerweise die Nummer "1".

Sie können den Steckplatz für die Schnittstellenbaugruppe angeben.

```
foreach(ConfigurationMode mode in configuration.Modes)
{
    Console.WriteLine("Mode name:{0}", mode.Name);
    foreach(ConfigurationPcInterface pcInterface in mode.PcInterfaces)
    {
        Console.WriteLine("PcInterface name:{0}", pcInterface.Name);
        Console.WriteLine("PcInterface number:{0}", pcInterface.Number);
        foreach(ConfigurationTargetInterface targetInterface in pcInterface.TargetInterfaces)
        {
            Console.WriteLine("TargetInterface:{0}", targetInterface.Name);
        }
    }
}
ConfigurationMode myMode = configuration.Modes.Find("PN/IE");
ConfigurationPcInterface myPcInterface = myMode.PcInterfaces.Find("PLCSIM", 1);
ConfigurationTargetInterface mySlot = myPcInterface.TargetInterfaces.Find("2 X3");
```

Hinweis

Wenn PLCSIM nicht läuft, ist es nicht in der PcInterfaces-Sammlung enthalten.

Sie können die Verbindung auch über die Adressen eines Subnetzes oder eines Gateways einrichten.

```
foreach(ConfigurationSubnet subnet in pcInterface.Subnets)
{
    Console.WriteLine("Subnet name:{0}", subnet.Name);
    foreach(ConfigurationAddress subnetAddress in subnet.Addresses)
    {
        Console.WriteLine("Subnet address:{0}", subnetAddress.Name);
    }
    foreach(ConfigurationGateway gateway in subnet.Gateways)
    {
        Console.WriteLine("Gateway name:{0}", gateway.Name);
        foreach(ConfigurationAddress gatewayAddress in gateway.Addresses)
        {
            Console.WriteLine("Gateway address:{0}", gatewayAddress.Name);
        }
    }
}
```

Diese Einstellungen können dann über den Aufruf von ApplyConfiguration im Objekt ConnectionConfiguration für die Online-Verbindung übernommen werden. Das System gibt

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

einen booleschen Wert zurück, der anzeigt, ob die Verbindung erfolgreich eingerichtet wurde.

```
bool result; result = configuration.ApplyConfiguration(slot);
//or
result = configuration.ApplyConfiguration(subnetAddress);
//or
result = configuration.ApplyConfiguration(gatewayAddress);
```

Nachdem die Verbindung eingerichtet ist, können Sie mit dem PLC in den Online-Betrieb gehen. Jeder Aufruf von `ApplyConfiguration` ist unabhängig und jeder nachfolgende Aufruf macht die zuvor übernommenen Einstellungen hinfällig. Die Einstellungen sind nur für die aktuelle Sitzung gültig, ähnlich wie in der Benutzeroberfläche. Wurden die Einstellungen in der Benutzeroberfläche bereits konfiguriert, ist kein Aufruf von `ApplyConfiguration` erforderlich und die Online-Verbindung nutzt die bereits verfügbaren Einstellungen.

Ist ein Gerät bereits online, wird beim Aufrufen von `ApplyConfiguration` eine Ausnahme ausgelöst.

Ob eine PLC bereits konfiguriert ist, können Sie mit der Eigenschaft `IsConfigured` im `ConnectionConfiguration`-Ziel feststellen:

```
bool isConfigured = configuration.IsConfigured;
```

PLCs mit nur einer möglichen Konfiguration sind immer konfiguriert, daher gibt Eigenschaft `IsConfigured` den Wert `true` zurück, ohne dass `ApplyConfiguration` aufgerufen wird.

Hinweis

Durch das Konfigurieren der Verbindung und die Online-Kommunikation wird der Inhalt von Offline- oder Online-PLCs nicht verändert. Der Benutzer muss entweder die Software und/oder Hardware auf den Online-PC laden oder die Software aus dem Online-Gerät hochladen, damit die PLCs konsistent sind.

Ältere Kommunikationseinstellungen

Ab TIA Portal Version 17 findet die Kommunikation von PLCs des Typs S7-1500 mit Firmware (FW) 2.9 über TLS statt.

Die Kommunikation ohne TLS ist mit `EnableLegacyCommunication` möglich.

EnableLegacyCommunication	Beschreibung
true	Kommunikation über TLS ist aktiviert
false	Kommunikation über TLS ist deaktiviert Dies ist die Kommunikation der TIA Portal Versionen < V17.

```
DeviceItem deviceItem = ...;
OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
ConnectionConfiguration configuration = onlineProvider.Configuration;
configuration.EnableLegacyCommunication = true;
```

Hinweis

Die Sicherheitserweiterung wird nur mit PLC-Firmware $\geq$ V2.9 (Hardwarekonfiguration) von PLC-Seite unterstützt.

Verbindungsparameter für die Kommunikation über TLS

Ab TIA Portal V17 sind PLCs des Typs S7-1500 mit Firmware (FW) 2.9 verfügbar. Diese PLCs unterstützen die sichere Kommunikation über TLS.

Für "Transport Layer Security" müssen bestimmte Bedingungen für die Kommunikation erfüllt sein. Sind diese Bedingungen nicht erfüllt, muss der Benutzer entscheiden, ob die Kommunikation vertrauenswürdig ist oder nicht.

Hierfür kann ein Event Handler TLS-Benachrichtigungen im Objekt ConnectionConfiguration abonnieren. Ist eine Entscheidung des Benutzers erforderlich, ruft TIA Portal Openness dieses Delegate auf und der Client kann es verarbeiten.

Der Zugriff auf das Objekt 'ConnectionConfiguration' ist über den Konfigurationsnavigator des OnlineProvider-Dienstes möglich.

Ein Ereignis für TLS-Benachrichtigung abonnieren

Um ein Ereignis für TLS-Benachrichtigungen zu abonnieren, müssen Sie Ihren Event Handler auf OnlineLegitimation registrieren.

```
private OnlineConfigurationDelegate m_OnlineConfigurationDelegate = null;
...
ConnectionConfiguration OnlineConfiguration = ...
m_OnlineConfigurationDelegate = OnlineCallBackMethod;
OnlineConfiguration.OnlineLegitimation += m_OnlineConfigurationDelegate;
```

Abonnement eines Ereignisses für TLS-Benachrichtigungen löschen

Um das Abonnement eines Ereignisses für TLS-Benachrichtigungen zu löschen (OnlineLegitimation), müssen Sie Ihren Event Handler registrieren:

```
ConnectionConfiguration OnlineConfiguration = ...
m_OnlineConfigurationDelegate = OnlineCallBackMethod;
OnlineConfiguration.OnlineLegitimation -= m_OnlineConfigurationDelegate;
```

TlsVerificationConfiguration

Wird eine Online-Verbindung mit einer Plc1500 mit FW 2.9 oder höher hergestellt, muss der Benutzer möglicherweise entscheiden, ob die Verbindung unter den jeweiligen Umständen wirklich hergestellt werden soll. In diesem Fall wird der Delegate des registrierten Event Handlers im Clientprogramm aufgerufen. Der Parameter OnlineConfiguration muss dann auf "TlsVerificationConfiguration" überprüft werden.

Die möglichen Werte für TlsVerificationConfigurationSelection sind:

TlsVerificationConfigurationSelection	Beschreibung
NonVerified	Die Verbindung ist nicht verifiziert und nicht vertrauenswürdig
Trusted	Die Verbindung ist verifiziert und vertrauenswürdig
NonTrusted	Die Verbindung ist verifiziert und nicht vertrauenswürdig

```
private void OnlineCallBackMethod(OnlineConfiguration onlineConfiguration)
{
    TlsVerificationConfiguration verificationConfiguration = onlineConfiguration as
    TlsVerificationConfiguration;
    if (verificationConfiguration != null)
    {
        verificationConfiguration.CurrentSelection =
        TlsVerificationConfigurationSelection.Trusted;
        //To establish a connection the value Trusted is necessary.
    }
}
```

Bei allen anderen Werten ist keine Online-Verbindung möglich und es wird eine Ausnahme ausgelöst.

TlsVerificationConfiguration unterstützt ferner zwei Attribute:

Name der Methode	Beschreibung
PlcName	Name der PLC, die den Delegate aufruft.
VerificationInfo	Meldung für den Benutzer. Enthält den Grund für den Aufruf des Delegates.

Siehe auch

[Verbindung zum TIA Portal aufbauen \(Seite 84\)](#)

[Projekt öffnen \(Seite 131\)](#)

5.11.2.19 Modulbeschreibung aktualisieren

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

In TIA Portal Openness wird der Dienst `ModuleDescriptionUpdater` am Geräteelement dazu verwendet, die aktuelle Modulbeschreibung auf die neueste Version des Moduls zu aktualisieren. Um das Geräteelement abzurufen, das Inhaber der Dienstinstantz ist, können Sie die Eigenschaft `OwnedBy` verwenden.

Mithilfe des Attributs `CanUpdate` können Sie anzeigen, ob eine neue `ConfigObject`-Version für dieses Deviceitem verfügbar ist.

Attribut	Datentyp	Beschreibung
<code>CanUpdate</code>	<code>bool</code>	TRUE: Eine neue Version ist verfügbar FALSE: Es gibt keine neue Version

Sie können mit der Aktion `UpdateModuleDescription ( )` die `ConfigObject`-Version eines Deviceitem aktualisieren.

Aktion	Rückgabewert	Beschreibung
<code>UpdateModuleDescription</code>	<code>True</code>	Das Deviceitem ist aktuell: • Das Deviceitem wurde erfolgreich aktualisiert • Das Deviceitem war bereits aktuell
	<code>False</code>	Das Deviceitem ist nicht aktuell: • Das Deviceitem konnte nicht aktualisiert werden

Programmcode

```
DeviceItem deviceItem = ...;
var descriptionUpdater = deviceItem.GetService<ModuleDescriptionUpdater>();
if (descriptionUpdater != null)
{
    if (descriptionUpdater.CanUpdate) //e.g is update module version possible
    {
        bool result = descriptionUpdater.UpdateModuleDescription();
        .
        .
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.2.20 Unterstützung von UMAC auf PLC-Gerät

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Aktionen auf dem PLC-Gerät ausführen, das über User Management Access Control (UMAC) geschützt ist.

Eine Legitimation über den Event Handler OnlineLegitimation ist für alle Online-PLC-Funktionen erforderlich, die eine Legitimation erfordern und UMAC unterstützen. Dies ist der gleiche Event Handler, der auch für die Kommunikation über TLS verwendet wird. Für Informationen über den Event Handler OnlineLegitimation siehe Abschnitt "Ein Ereignis für TLS-Benachrichtigung abonnieren" Unterstützung der sicheren S7-Kommunikation mit TLS (Seite 492)

Um den Event Handler für die Online-Legitimation zu registrieren, benötigen Sie das Objekt der Verbindungskonfiguration. Sie können auf das Objekt der Verbindungskonfiguration über den Konfigurationsnavigator des entsprechenden Online-Diensts zugreifen (z. B. OnlineProvider, FingerprintDataProvider, DownloadProvider, StationUploadProvider usw.).

Programmcode: Auf das Objekt der Verbindungskonfiguration zugreifen

Um auf das Konfigurationsobjekt zuzugreifen, ändern Sie folgenden Programmcode:

```
...
DeviceItem deviceItem = ...;
DownloadProvider downloadProvider = deviceItem.GetService<DownloadProvider>();
ConnectionConfiguration onlineConfiguration = downloadProvider.Configuration;
...
```

Programmcode: Ein Ereignis für Online-Legitimation abonnieren

Um ein Ereignis für die Benachrichtigungen zur Online-Legitimation zu abonnieren, müssen Sie Ihren Event Handler auf OnlineLegitimation registrieren:

```

ConnectionConfiguration onlineConfiguration = ...;
OnlineConfigurationDelegate onlineConfigurationDelegate = OnlineCallBackMethod;
onlineConfiguration.OnlineLegitimation += onlineConfigurationDelegate;
...
private void OnlineCallBackMethod(OnlineConfiguration onlineConfiguration)
{
...

```

Hinweis

Die Registrierung für dieses Ereignis ist nur erforderlich, wenn sie nicht im Rahmen der Tls-Kommunikation erfolgt ist.

Programmcode: Ein Abonnement für ein Ereignis für Online-Legitimation löschen

Um das Abonnement eines Ereignisses für Online-Benachrichtigungen (OnlineLegitimation) zu löschen, müssen Sie Ihren Event Handler aus dem Register löschen:

```

ConnectionConfiguration onlineConfiguration = ...;
OnlineConfigurationDelegate onlineConfigurationDelegate = OnlineCallBackMethod;
onlineConfiguration.OnlineLegitimation -= onlineConfigurationDelegate;
...
private void OnlineCallBackMethod(OnlineConfiguration onlineConfiguration)
{
..

```

OnlineAuthenticationConfiguration

OnlineAuthenticationConfiguration hat die folgenden Eigenschaften und Methoden:

OnlineAuthenticationConfiguration	Beschreibung
bool IsSecureCommunication { get; }	Das Attribut liefert "true" im Falle einer "Sicheren Kommunikations"-Verbindung (TLS-Handshake) zur PLC. In allen anderen Fällen liefert das Attribut "false". Weitere Informationen finden Sie unter Herunterladen zu den PLCs (Seite 400)
IList<AuthenticationType> GetSupportedAuthenticationTypes()	Abrufen der unterstützten AuthenticationTypes über GetSupportedAuthenticationTypes() von der PLC.
OnlineCredentials OnlineCredentials { get; }	Eigenschaft zur Konfiguration der Einzelheiten der Anmeldedaten.

IsSecureCommunication

Im Objekt `OnlineAuthenticationConfiguration` können Sie mit der Option `IsSecureCommunication` überprüfen, ob die Verbindung sichere Tls verwendet. Dies ist mit der Konfiguration für den Upload oder Download über `DownloadConfiguration` vergleichbar.

```
OnlineAuthenticationConfiguration onlineAuthenticationConfiguration = onlineConfiguration
as OnlineAuthenticationConfiguration;
if (onlineAuthenticationConfiguration != null)
{
    // Secure communication via TLS?
    if (onlineAuthenticationConfiguration.IsSecureCommunication)
    {
        ...
    }
}
```

GetSupportedAuthenticationTypes() und AuthenticationType

GetSupportedAuthenticationTypes()

Je nach der heruntergeladenen UMAC-Konfiguration sind verschiedene Typen der Authentifizierungslegitimation möglich. Um die unterstützten Typen abzurufen, müssen Sie:

```
OnlineAuthenticationConfiguration onlineAuthenticationConfiguration = onlineConfiguration
as OnlineAuthenticationConfiguration;
if (onlineAuthenticationConfiguration != null)
{
    IList<AuthenticationType> supportedTypes =
onlineAuthenticationConfiguration.GetSupportedAuthenticationTypes();
    foreach (AuthenticationType supportedType in supportedTypes)
    {
        ...
    }
}
```


AuthenticationType

AuthenticationType stellt den Typ der Authentifizierung dar, den Sie wählen können. Um den Typ der Authentifizierung abzurufen, können Sie den UserType-Wert des UserType-Attributs des CurrentUserType-Attributs abrufen.

AuthenticationType	Attribut	Beschreibung
UserType CurrentUserType { get; }	Liefert den UserType, unterstützt von der PLC	Dieser Wert kann in OnlineCredentials verwendet werden: UserType Die möglichen Einträge für UserType für die von der PLC unterstützen Legitimation sind: • Keine - Keine Legitimation erforderlich • AnonymousUser - Legitimation als anonymer Benutzer • GlobalUser - Legitimation als globaler Benutzer • ProjectUser - Legitimation als Projektbenutzer • SingleSignOnUser - Benutzer mit Einmalanmeldung • PasswordOnly - Alte Legitimation mit Passwort für den PLC-Schutz der Zugriffsstufe

```

...
IList<AuthenticationType> supportedTypes =
onlineAuthenticationConfiguration.GetSupportedAuthenticationTypes();
foreach (AuthenticationType supportedType in supportedTypes)
{
    UserType userType = supportedType.CurrentUserType;
    if (userType == UserType.ProjectUser)
    {
        ...
    }
}
...

```

OnlineCredentials

OnlineCredentials hat die folgenden Eigenschaften und Methoden:

OnlineCredentials	Aktion	Beschreibung
void SetPassword(SecureString password)	Einstellung des Passworts über die SetPassword-Methode.	Geben Sie ein Passwort in die OnlineCredentials ein, um Zugriff auf das Modul zu erhalten.
string Name { get; set; }	UserName über die Eigenschaft Name empfangen oder angeben.	Geben Sie den Namen als UserName in die OnlineCredentials ein.
UserType Type { get; set; }	UserType über die Eigenschaft Type empfangen oder angeben	Geben Sie UserType in die OnlineCredentials ein. Die möglichen Werte sind verfügbar über GetSupportedAuthenticationTypes()

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Nachdem Sie die unterstützten Benutzertypen und die notwendigen Legitimationsinformationen überprüft haben, müssen Sie diese in den OnlineCredentials einstellen:

```
OnlineAuthenticationConfiguration onlineAuthenticationConfiguration = onlineConfiguration
as OnlineAuthenticationConfiguration;
if (onlineAuthenticationConfiguration != null)
{
    // Check supported user types and legitimation information
    ...
    // Set the online credentials for legitimation
    OnlineCredentials onlineCredentials =
onlineAuthenticationConfiguration.OnlineCredentials;
    SecureString password = ...;
    onlineCredentials.Type = UserType.ProjectUser;
    onlineCredentials.Name = "User1";
    onlineCredentials.SetPassword(password);
}
```

Hinweis

Die explizite Legitimation ist innerhalb des Rückrufs nicht erforderlich. Die Legitimation erfolgt nach der Rückgabe des Rückrufs. Bei unzureichenden Zugriffsrechten wird die Online-Aktion (z. B. Download) mit einer Ausnahme einer unzulässigen Verbindung abgebrochen.

5.11.2.21 Zugänglichkeit

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 131)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die IP-Zugänglichkeit unterstützen. In TIA Portal Openness wird an der Geräteelement-CPU ein neues dynamisches Attribut `PlcAccessCommunicationModule` verfügbar. Das Attribut gehört zum Typ Objekt. Dadurch können Sie auf das CP-Objekt zugreifen, das Sie für die IP-Zugänglichkeit auswählen möchten, und das Objekt dem `PlcAccessCommunicationModule` zuordnen.

Wenn bereits ein CP ausgewählt wurde, können Sie das CP-Objekt abrufen, indem Sie den Wert von `PlcAccessCommunicationModule` abrufen.

Programmcode

```
DeviceItem Plc = ...;  
object communicationModule = Plc.GetAttribute("PlcAccessCommunicationModule");  
// cP contains the object reference of a plugged CP  
Plc.SetAttribute("PlcAccessCommunicationModule", cP);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.2.22 Zugreifen auf den Konfigurationsanbieter für die PLC-Zugriffskontrolle

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit dem Dienst `PlcAccessControlConfigurationProvider` in TIA Portal Openness können Sie die Konfiguration der Zugriffskontrolle während der Runtime aktivieren und deaktivieren.

Methode, Parameter und Enum

Methode

Der Dienst `PlcAccessControlConfigurationProvider` stellt ein Attribut namens `PlcAccessControlConfiguration` zur Verfügung, das die folgenden Funktionen zur Konfiguration der Zugriffskontrolle unterstützt.

Methodenname	Rückgabe des Datentyps	Beschreibung
<code>GetAttribute(string attributeName)</code>	integer	Verwendet zum Abruf von <code>PlcAccessControlConfiguration</code>
<code>SetAttribute(string attributeName, int configuration)</code>	void	Verwendet zur Einstellung von <code>PlcAccessControlConfiguration</code>

Hinweis

Alle S7-1500 PLCs ab Firmware-Version 3.1 unterstützen `PlcAccessControlConfiguration`.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Parameter

Die Methode `GetAttribute()` unterstützt den folgenden Parameter zum Abruf von `PlcAccessControlConfiguration`:

Parametername	Datentyp	Beschreibung
attributeName	string	Gibt den Namen des Parameters an, zum Beispiel "PlcAccessControlConfiguration" in diesem Fall. Die Enum-Elemente können zurückgegeben werden: - Disabled - EnabledWithAccessControl - EnabledWithAccessControlViaAccessLevel

Die Methode `SetAttribute()` unterstützt den folgenden Parameter zur Einstellung von `PlcAccessControlConfiguration`:

Parametername	Datentyp	Beschreibung
attributeName	string	Gibt den Namen des Parameters an, zum Beispiel "PlcAccessControlConfiguration" in diesem Fall
configuration	Die Enum-Elemente können zurückgegeben werden: - Disabled - EnabledWithAccessControl - EnabledWithAccessControlViaAccessLevel	Gültige Konfiguration für die PLC-Zugriffskontrolle. Die Rückgabe für ungültige Szenarien ist leer. Diese werden über eine Ausnahme verarbeitet.

Enum

`PlcAccessControlConfiguration` stellt die folgenden Enum-Werte und die Beschreibung zur Verfügung:

Enum	Wert	Beschreibung
Disabled	0	Keine Änderung beim Zugriff auf die PLC über das automatisierte System.
EnabledWithAccessControl	2	Es gibt keine separate Konfiguration für verschiedene Zugriffsmodi auf die PLC wie (AccessLevel, WebServer, OPCUA).
EnabledWithAccessControlViaAccessLevel	3	Es gibt keine separate Konfiguration für verschiedene Zugriffsmodi auf die PLC wie (WebServer, OPCUA), Sie können jedoch auf die PLC über Passwörter auf der Zugriffsstufe zugreifen.

Programmcode: GetAttribute()

```
// Example to use 'GetAttribute'
PlcAccessControlConfigurationProvider provider =
plc.GetService<PlcAccessControlConfigurationProvider>();
var configuration = provider.GetAttribute("PlcAccessControlConfiguration");
```

Programmcode: SetAttribute()

```
// Example to use 'SetAttribute'
PlcAccessControlConfigurationProvider provider =
plc.GetService<PlcAccessControlConfigurationProvider>();
provider.SetAttribute("PlcAccessControlConfiguration",
PlcAccessControlConfiguration.EnabledWithAccessControlViaAccessLevel);
```

5.11.2.23 Display-Passwort festlegen**Voraussetzung**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Sie können mit TIA Portal Openness das Passwort zum Schutz des Displays festlegen. Sie können mit der Methode `SetPassword(SecureString password)` das Passwort zum Schutz des Displays festlegen, wobei der Parameter für das Passwort ein `SecureString` sein muss. Das Bestätigungspasswort wird in Openness nicht benötigt und ist daher nicht verfügbar.

Die Funktion ist verfügbar, wenn die folgenden Voraussetzungen erfüllt sind:

- Die PLC verfügt über eine Instanz "Display".
- Das "Display" unterstützt die Funktion zum Schutz durch ein "Passwort". Es gibt Fälle, z. B. eine Software-PLC, in denen "Display" verfügbar, aber kein Passwortschutz verfügbar ist.

Programmcode

Ändern Sie folgenden Programmcode, um das Geräteelement abzurufen, das Inhaber der Dienstinstanz ist. Hierzu kann die Eigenschaft `OwnedBy` verwendet werden.

```
DisplayProtection displayProtection = plcDisplay.GetService<DisplayProtection>();
var displayDeviceItem = displayProtection.OwnedBy;
```

Ändern Sie folgenden Programmcode, um das Passwort zum Schutz des Displays festzulegen:

```
displayProtection.SetPassword(someSecuredString);
//Sets the string someSecuredString as the CPU display protection password.
```

Hinweis

Software-Controller verfügen über ein Display-Submodul, unterstützen jedoch keinen Displayschutz. Deshalb ist in TIA Portal Openness die Funktion SetPassword am Display-Submodul von Software-Controllern nicht verfügbar.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.2.24 Zugriff auf Webserver und OPC UA-Benutzerverwaltung

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Sie können mit TIA Portal Openness auf den Webserver und das OPC UA-Submodul von PLCs zugreifen. Es ist möglich, einen Benutzer bis zur Höchstzahl am Webserver und am OPC UA-Submodul von PLCs hinzuzufügen. Dieser Benutzer hat einen Benutzernamen und ein Passwort.

Benutzerverwaltung am Webserver

Sie können die folgenden Vorgänge wie z. B. das Hinzufügen eines Benutzers, das Löschen eines Benutzers und das Festlegen des Passworts am Webserver für alle unterstützten Geräte nur dann durchführen, wenn der Webserver am Modul aktiviert ist.

Wenn der Webserver nicht aktiviert ist, wird durch das Hinzufügen und Löschen von Benutzern und das Festlegen des Passworts eine `EngineeringTargetInvocationException` ausgelöst. Die Lesevorgänge sind trotzdem verfügbar.

Der Dienst `WebServerUserManagement` ist für die PLC-Instanz nur der unterstützten Geräte verfügbar. Für alle nicht unterstützten Geräte lautet der Dienst null.

```
WebServerUserManagement
{
  Navigators WebServerUsers
  {
    WebServerUserComposition
  }
}
```

Der Dienst stellt einen Navigator namens `WebServerUsers` zur Verfügung, mit dessen Hilfe eine `WebServerUserComposition` bezogen werden kann, die die Instanzen von `WebServerUser` verwaltet.

```
WebServerUserComposition
{
WebServerUser Find(string username);
void Create(string username, WebserverUserPermissions permissions, SecureString password);
}
WebServerUser
{
string UserName{get;}
WebserverUserPermissions Permissions {get;set;}
void Delete();
void SetPassword(SecureString password);
}
```

Programmcode: Aktionen an der `WebServerUserComposition`

Ändern Sie das folgende Programm, um einen Webserver-Benutzer zu suchen:

```
WebServerUserComposition webServerUserComposition = WebServerUserManagement.WebServerUsers;
WebServerUser user1 = webServerUserComposition.Find("user1");
```

Ändern Sie folgenden Programmcode, um einen neuen Webserver-Benutzer mit Berechtigungen für Webserver-Benutzer zu erstellen:

```
WebServerUserComposition webServerUserComposition = WebServerUserManagement.WebServerUsers;
WebServerUser user1 = webServerUserComposition.Create("user1",
WebserverUserPermissions.ReadTagStatus, someSecureString);
```

Ändern Sie folgenden Programmcode, um den Webserver-Benutzer zu löschen:

```
WebServerUserComposition webServerUserComposition = WebServerUserManagement.WebServerUsers;
WebServerUser user1 = webServerUserComposition.Find("user1");
user1.Delete();
```

Ändern Sie folgenden Programmcode, um das aktuelle Passwort des Benutzers durch das neue Passwort zu ersetzen:

```
WebServerUserComposition webServerUserComposition = WebServerUserManagement.WebServerUsers;
WebServerUser user1 = webServerUserComposition.Find("user1");
user1.SetPassword(someSecureString);
```

Benutzerverwaltung am OPC UA-Server

Sie können die folgenden Vorgänge wie z. B. das Hinzufügen eines Benutzers, das Löschen eines Benutzers und das Festlegen des Passworts am OPC UA-Server nur dann durchführen, wenn der OPC UA-Server aktiviert ist und der Benutzername und die Passwort-Authentifizierung aktiviert sind.

Wenn diese Voraussetzungen nicht erfüllt sind, wird durch den Vorgang eine `EngineeringTargetInvocationException` ausgelöst, deren Fehlerdetails besagen, dass die Authentifizierung aktiviert werden muss.

Der Dienst `OpcUaUserManagement` ist nur am OPC UA-Submodul der PLC verfügbar. Der Dienst ist für alle unterstützten Geräte verfügbar, bei denen das OPC UA-Submodul für das Geräteelement verfügbar ist. Andernfalls wird der Dienst mit dem Wert null zurückgegeben.

```
OpcUaUserManagement
{
  Navigators OpcUaUsers
  {
    OpcUaUserComposition
  }
}
```

Der Dienst stellt einen Navigator namens `OpcUaUsers` zur Verfügung, mit dessen Hilfe eine `OpcUaUserComposition` bezogen werden kann. Die Zusammensetzung verwaltet die Instanzen von `OpcUaUser`:

```
OpcUaUserComposition
{
  OpcUaUser Find(string username);
  void Create(string username, SecureString password);
}
OpcUaUser
{
  string UserName{get;}
  void Delete();
  void SetPassword(SecureString password);
}
```

Programmcode: Aktionen an der OpcUaUserComposition

Ändern Sie folgenden Programmcode, um einen `OpcUaUser` zu suchen:

```
OpcUaUserComposition opcUaUserComposition = opcUaUserManagement.OpcUaUsers;
OpcUaUser user1 = opcUaUserComposition.Find("user1");
```

Ändern Sie folgenden Programmcode, um einen neuen `OpcUaUser` zu erstellen:

```
OpcUaUserComposition opcUaUserComposition = opcUaUserManagement.OpcUaUsers;
opcUaUserComposition.Create("user1", someSecureString);
```


Ändern Sie folgenden Programmcode, um den OpcUaUser zu löschen:

```
OpcUaUserComposition opcUaUserComposition = opcUaUserManagement.OpcUaUsers;
OpcUaUser user = opcUaUserComposition.Find("user1");
user.Delete();
```

Ändern und verwenden Sie folgenden Programmcode, um das aktuelle Passwort durch ein bereitgestelltes zu ersetzen:

```
OpcUaUserComposition opcUaUserComposition = opcUaUserManagement.OpcUaUsers;
OpcUaUser user = opcUaUserComposition.Find("user1");
user.SetPassword(someSecureString);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.3 Bausteine

5.11.3.1 Gruppe "Programmbausteine" abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.

Programmcode

Ändern Sie den folgenden Programmcode, um die Gruppe "Programmbausteine" abzufragen:

```
private static void GetBlockGroupOfPLC(PlcSoftware plcsoftware)
//Retrieves the system group of a block
{
    PlcBlockSystemGroup blockGroup = plcsoftware.BlockGroup;
}
```

5.11.3.2 Systemgruppe für Systembausteine abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode:

Ändern Sie den folgenden Programmcode, um die für Systembausteine vom System erzeugte Gruppe zu ermitteln:

```
PlcSoftware plcSoftware = ...
foreach (PlcSystemBlockGroup systemGroup in plcSoftware.BlockGroup.SystemBlockGroups)
{
    foreach (PlcSystemBlockGroup group in systemGroup.Groups)
    {
        PlcBlockComposition pbComposition = group.Blocks;
        foreach (PlcBlock block in pbComposition)
        {
            //Fügen Sie Ihren Code hier hinzu
        }
    }
}
```

5.11.3.3 Systemuntergruppen enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode: Alle Systemuntergruppen enumerieren

Ändern Sie den folgenden Programmcode, um die Systemuntergruppen aller Systembausteine zu enumerieren:

```
//Retrieves the system generated group for system blocks
private static void GetSystemgroupForSystemblocks(PlcSoftware plcSoftware)
{
    PlcSystemBlockGroupComposition systemBlockGroups =
plcSoftware.BlockGroup.SystemBlockGroups;
    if (systemBlockGroups.Count != 0)
    {
        PlcSystemBlockGroup sbSystemGroup = systemBlockGroups[0];
        foreach (PlcSystemBlockGroup group in sbSystemGroup.Groups)
        {
            EnumerateSystemBlockGroups(group);
        }
    }
}
private static void EnumerateSystemBlockGroups(PlcSystemBlockGroup systemBlockGroup)
{
    foreach (PlcSystemBlockGroup group in systemBlockGroup.Groups)
    {
        // recursion EnumerateSystemBlockGroups(group);
    }
}
```

Programmcode: Auf eine bestimmte Untergruppe zugreifen

Ändern Sie den folgenden Programmcode, um auf eine bestimmte Untergruppe zuzugreifen:

```
private static void AccessSbGroup(PlcSystemBlockGroup systemBlockGroup)
{
    PlcSystemBlockGroup group1 = systemBlockGroup.Groups.Find("User group XYZ");
    PlcSystemBlockGroup group2 = group1.Groups.Find("User group ZYX");
}
```

Siehe auch

Externe Datei hinzufügen (Seite 528)

5.11.3.4 Benutzerdefinierte Bausteingruppen enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.

Verwendung

Enthaltene Untergruppen werden beim Enumerieren rekursiv berücksichtigt.

Programmcode: Alle Gruppen enumerieren

Ändern Sie den folgenden Programmcode, um die benutzerdefinierten Bausteingruppen zu enumerieren:

```
//Enumerates all block user groups including sub groups
private static void EnumerateAllBlockGroupsAndSubgroups(PlcSoftware plcsoftware)
{
    foreach (PlcBlockUserGroup blockUserGroup in plcsoftware.BlockGroup.Groups)
    {
        EnumerateBlockUserGroups(blockUserGroup);
    }
}

private static void EnumerateBlockUserGroups(PlcBlockUserGroup blockUserGroup)
{
    foreach (PlcBlockUserGroup subBlockUserGroup in blockUserGroup.Groups)
    {
        EnumerateBlockUserGroups(subBlockUserGroup);
        // recursion
    }
}
```

Programmcode: Auf eine Gruppe zugreifen

Ändern Sie den folgenden Programmcode, um auf eine ausgewählte benutzerdefinierte Bausteingruppe zuzugreifen:

```
//Gives individual access to a specific block user group
private static void AccessBlockusergroup(PlcSoftware plcsoftware)
{
    PlcBlockUserGroupComposition userGroupComposition = plcsoftware.BlockGroup.Groups;
    PlcBlockUserGroup plcBlockUserGroup = userGroupComposition.Find("MyUserfolder");
}
```

5.11.3.5 Alle Bausteine enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.

Verwendung

Der gezielte Zugriff auf einen Programmbaustein ist möglich, wenn dessen Name bekannt ist.

Programmcode: Alle Bausteine enumerieren

Ändern Sie den folgenden Programmcode, um die Bausteine aller Bausteingruppen zu enumerieren:

```
private static void EnumerateAllBlocks(PlcSoftware plcsoftware)
//Enumerates all blocks
{
    foreach (PlcBlock block in plcsoftware.BlockGroup.Blocks)
    {
        // Do something...
    }
}
```

Programmcode: Auf einen bestimmten Baustein zugreifen

Um auf einen bestimmten Baustein zuzugreifen, ändern Sie folgenden Programmcode:

```
private static void AccessASingleBlock(PlcSoftware plcsoftware)
//Gives individual access to a block
{
    // The parameter specifies the name of the block
    PlcBlock block = plcsoftware.BlockGroup.Blocks.Find("MyBlock");
}
```

5.11.3.6 Informationen eines Bausteins/Anwenderdatentyps abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die TIA Portal Openness API unterstützt die Abfrage der folgenden Informationen für Programm und Datenbausteine und für Anwenderdatentypen:

- Zeitstempel im UTC-Zeitformat
Über den Zeitstempel erfahren Sie Folgendes:
 - Wann der Baustein zuletzt übersetzt wurde
 - Wann der Baustein zuletzt geändert wurde
- Attribut „Consistency“
Das Attribut „Consistency“ ist in folgenden Fällen auf „True“ gesetzt:
 - Der Baustein wurde erfolgreich übersetzt.
 - Der Baustein wurde seit der Übersetzung nicht geändert.
 - An externen Objekten wurden keine Änderungen vorgenommen, die eine Neuübersetzung erforderlich machen würden.
- Verwendete Programmiersprache (nur Programm und Datenbausteine)
- Bausteinnummer
- Bausteinname
- Bausteinautor
- Bausteinfamilie
- Bausteintitel
- Bausteinversion

Weitere Informationen hierzu siehe Bausteine und Typen des TIA Portal Openness-Objektmodells (Seite 61).

Programmcode

Um die oben aufgeführten Informationen abzufragen, ändern Sie folgenden Programmcode:

```
private static void GetPlcBlockInformation(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    // Read information
    DateTime compileDate = plcBlock.CompileDate;
    DateTime modifiedDate = plcBlock.ModifiedDate;
    bool isConsistent = plcBlock.IsConsistent;
    int blockNumber = plcBlock.Number;
    string blockName = plcBlock.Name;
    ProgrammingLanguage programmingLanguage = plcBlock.ProgrammingLanguage;
    string blockAuthor = plcBlock.HeaderAuthor;
    string blockFamily = plcBlock.HeaderFamily;
    string blockTitle = plcBlock.HeaderName;
    System.Version blockVersion = plcBlock.HeaderVersion;
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.11.3.7 Einen Baustein schützen und Schutz aufheben

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Verwendung

Mit TIA Portal Openness können Sie über die Klasse PlcBlockProtectionProvider und den Dienst PlcBlockProtectionProvider einen Baustein mit einem Passwort schützen und den Passwortschutz wieder aufheben. Der Dienst PlcBlockProtectionProvider ist bei Bausteinen verfügbar, die folgende Voraussetzungen erfüllen:

- Der Baustein kann einen Know-how-Schutz erhalten
- Der Baustein ist ein Code-Baustein oder eine globale DB
- Der Baustein muss übersetzt werden
- Der Baustein ist im aktuellen PLC unterstützt oder editierbar
- Der Baustein befindet sich nicht in einem schreibgeschützten Kontext
- Der Baustein ist nicht Know-how-geschützt

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

- Der Baustein ist nicht online
- Der Baustein ist keine CPU-DB
- Der Baustein nutzt keine klassische Verschlüsselungssprache, ProDiag oder ProDiag-OB
- Der Baustein ist kein verschlüsselter importierter klassischer Baustein

Falls der Baustein nicht alle Bedingungen erfüllt, wird von der Methode `GetService()` eine Nullreferenz ausgegeben.

Einen Baustein schützen

Legen Sie mit der Methode `Protect()` ein Passwort fest, um den Programmierbaustein zu schützen.

```
void Protect(SecureString password)
```

In folgenden Fällen tritt ein Fehler auf:

- Wenn Sie einen bereits geschützten Baustein zu schützen versuchen: Eine `EngineeringTargetInvocationException` wird mit einer Meldung ausgelöst, die besagt, dass Sie kein bereits geschütztes Objekt schützen können.
- Wenn Sie einen Baustein mit einem leeren String als Passwort zu schützen versuchen: Eine `EngineeringTargetInvocationException` wird mit einer Meldung ausgelöst, die besagt, dass kein Passwort angegeben wurde.
- Wenn Sie einen fehlersicheren Baustein zu schützen versuchen, während das Failsafe-Programm passwortgeschützt ist: Eine `EngineeringTargetInvocationException` wird ausgelöst.
- Wenn Sie einen fehlersicheren Baustein zu schützen versuchen, während dieser Baustein nicht aufgerufen ist: Eine `EngineeringTargetInvocationException` wird ausgelöst.

Ungeschützter Baustein

Entfernen Sie das Passwort, mit dem der Programmierbaustein geschützt ist, mit Hilfe der Methode `Unprotect()`.

```
void Unprotect(SecureString password)
```

In folgenden Fällen tritt ein Fehler auf:

- Wenn Sie den Schutz eines bereits ungeschützten Bausteins aufzuheben versuchen: Eine `EngineeringTargetInvocationException` wird mit einer Meldung ausgelöst, die besagt, dass Sie nicht den Schutz eines ungeschützten Objekts aufheben können.
- Wenn Sie den Schutz eines Bausteins mit dem falschen Passwort aufzuheben versuchen: Eine `EngineeringTargetInvocationException` wird mit einer Meldung ausgelöst, die besagt, dass das Passwort nicht akzeptiert wurde.
- Wenn Sie einen Baustein mit einem leeren String als Passwort zu schützen versuchen: Eine `EngineeringTargetInvocationException` wird mit einer Meldung ausgelöst, die besagt, dass kein Passwort angegeben wurde.

Programmcode

```
PlcBlock block = ...;
PlcBlockProtectionProvider protectionProvider =
block.GetService<PlcBlockProtectionProvider>();
if (protectionProvider != null)
{
    // ...
    // perform know-how protection related operations here
}
```

Nach ungültigen Zeichen suchen

Sie können einen Baustein unter Verwendung der Protect()-Methode mit beliebigen Zeichen schützen, einschließlich Rückschritt, Tab usw. Bei dem Passwort, das zum Schutz eines Bausteins angegeben wird, handelt es sich um einen SecureString. Sie können mit GetInvalidPasswordCharacters() prüfen, ob das angegebene Passwort ungültige Zeichen enthält.

```
SecureString CreatePasswordString(ProtectionProvider protectionProvider, IEnumerable<char>
contentCharacters)
{
    IList<char> invalidCharacters = protectionProvider.GetInvalidPasswordCharacters();
    SecureString password = new SecureString();
    foreach(char ch in contentCharacters)
    {
        if (!invalidCharacters.Contains(ch))
        {
            password.AppendChar(ch);
        }
        else
        {
            // at least one of the content characters is not valid
            // signal an error - e.g. throw an exception
            ...
        }
    }
    return password;
}
```

In folgenden Fällen tritt ein Fehler auf:

- Wenn Sie den Schutz eines bereits ungeschützten Bausteins aufzuheben versuchen: Eine `EngineeringTargetInvocationException` wird mit einer Meldung ausgelöst, die besagt, dass Sie nicht den Schutz eines ungeschützten Objekts aufheben können.
- Wenn Sie den Schutz eines Bausteins mit dem falschen Passwort aufzuheben versuchen: Eine `EngineeringTargetInvocationException` wird mit einer Meldung ausgelöst, die besagt, dass das Passwort nicht akzeptiert wurde.
- Wenn Sie einen Baustein mit einem leeren String als Passwort zu schützen versuchen: Eine `EngineeringTargetInvocationException` wird mit einer Meldung ausgelöst, die besagt, dass kein Passwort angegeben wurde.

5.11.3.8 Baustein löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Programmcode

Ändern Sie den folgenden Programmcode, um einen Baustein zu löschen:

```
//Runs through block group and deletes blocks
private static void DeleteBlocks(PlcSoftware plcsoftware)
{
    PlcBlockSystemGroup group = plcsoftware.BlockGroup;
    // or BlockUserGroup group = ...;
    for (int i = group.Blocks.Count - 1; i >= 0; i--)
    {
        PlcBlock block = group.Blocks[i];
        if (block != null)
        {
            block.Delete();
        }
    }
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.11.3.9 Gruppe für Bausteine erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um eine Gruppe für Bausteine zu erzeugen:

```
private static void CreateBlockGroup(PlcSoftware plcsoftware)
//Creates a block group
{
    PlcBlockSystemGroup systemGroup = plcsoftware.BlockGroup;
    PlcBlockUserGroupComposition groupComposition = systemGroup.Groups;
    PlcBlockUserGroup myCreatedGroup = groupComposition.Create("MySubGroupName");
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.11.3.10 Gruppe für Bausteine löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Programmcode

Um eine Gruppe für Bausteine zu löschen, ändern Sie folgenden Programmcode:

```
// Deletes user groups from PlcBlockSystemGroup or PlcBlockUserGroup
private static void DeleteBlockFolder(PlcSoftware plcSoftware)
{
    PlcBlockUserGroup group = plcSoftware.BlockGroup.Groups.Find("myGroup");
    //PlcBlockSystemGroup group = plcSoftware.BlockGroup;
    PlcBlockUserGroupComposition subgroups = group.Groups;
    PlcBlockUserGroup subgroup = subgroups.Find("myUserGroup");
    if (subgroup != null)
    {
        subgroup.Delete();
    }
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.11.3.11 Auf Attribute sämtlicher Bausteine zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können in TIA Portal Openness die für alle Bausteine geltenden Attribute mit der Methode `SetAttribute()` und `SetAttributes()` festlegen.

Die Methode `SetAttributes()` kann für alle Objekttypen verwendet werden, bei denen ein schreibbares Openness-Attribut verfügbar ist. Bei einem Eingabeelement verhält es sich wie `SetAttribute`.

Parameterprüfung:

- Anfängliche Prüfungen werden durchgeführt, bevor das erste Element von der globalen PE-Infrastruktur verarbeitet wird: Die Eingabeelemente werden auf identische und gültige Attributnamen sowie auf Übereinstimmung der Datentypen geprüft. Ein Attribut kann von einem `SetAttributes` nicht zweimal festgelegt werden.
- Eine sequenzielle Prüfung erfolgt während der Ausführung von `SetAttributes`. Die Reihenfolge der Verarbeitung entspricht der Reihenfolge der Elemente in der Eingabeliste.

Die Methode `SetAttributes` wird in nur einer Transaktion verarbeitet. Wenn ein Attribut der Eingabeliste nicht festgelegt werden kann, bleiben die Werte der Attribute unverändert. `SetAttributes` legt die Attribute in der gleichen Reihenfolge fest, wie sie in der Parameterliste

vorhanden sind. Die Parameter können voneinander abhängig oder unabhängig sein. Wenn ein Parameter von einem vorherigen Parameter in der Liste abhängig ist, wird dieser Parameter geprüft und entsprechend dem aktuellen Wert des anderen Attributs ausgewertet (nicht entsprechend dem ursprünglichen Wert zum Zeitpunkt des Aufrufs). Wenn SetAttributes nicht ausgeführt werden kann, werden Einzelheiten zum Fehler angegeben. Die Fehlermeldung enthält den Namen des ersten Attributs, das nicht festgelegt werden konnte, und dessen Wert sowie den Grund dafür.

Die folgenden Programmcodebeispiele basieren auf den zwei Attributen AutoNumber und Number, die SetAttribute() verwenden (siehe Bausteine exportieren (Seite 1395) für alle geltenden Attribute von Bausteinen).

Programmcode: SetAttribute

```
...
PlcBlockGroup blockFolder = YourUtilities.GetFolder();
var block = blockFolder.Blocks.Find("Block_1");
if ((bool)block.GetAttribute("AutoNumber")==true)
{
    block.SetAttribute("AutoNumber", false);
}
block.SetAttribute("Number", 2);
...
```

Programmcode: SetAttributes

```
PlcBlock block = SelectBlock("MC-Servo");
if (block != null)
{
    IList<KeyValuePair<string, object>> list = new List<KeyValuePair<string, object>>()
    {
        new KeyValuePair<string, object>("DataExchangeMode", OBDataExchangeMode.Synchronous), new
        KeyValuePair<string, object>("SynchronousApplicationCycleTime", (float)69)
    };
    try
    {
        block.SetAttributes(list);
    }
    catch (EngineeringException e)
    {
        Console.WriteLine("Exception: " + e.Message);
    }
}
```

5.11.3.12 Einen ProDiag-FB anlegen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit der Create-Aktion der PLCBlock-Zusammensetzung einen ProDiag-FB mit den folgenden Parametern anlegen.

1. Name
2. Auto-Nummerierungsmerker
3. Nummer
 - Wenn "Auto-Nummerierungsmerker" wahr ist, wird die angegebene Bausteinnummer festgelegt, sofern sie frei ist. Andernfalls wird ein neuer Baustein generiert.
 - Wenn "Auto-Nummerierungsmerker" falsch ist, wird für die angegebene Bausteinnummer der Baustein festgelegt.
4. Programmiersprache
 - Wenn Sie eine Create-Aktion mit der Programmiersprache ProDiag aufrufen, wird ein neuer FB ohne IDB angelegt.
 - Wenn Sie die Create-Aktion mit IDB von ProDiag aufrufen, wird IDB von ProDiag angelegt.
 - In jedem anderen nicht unterstützten Fall wird eine wiederherstellbare Ausnahme ausgelöst.

Programmcode: Einen ProDiag-FB anlegen

```
PlcSoftware plc = ...;
PlcBlockGroup blockFolder = plc.BlockGroup;
PlcBlockComposition blockComposition = blockFolder.Blocks;
if (blockComposition != null)
{
    string fbName = "ProDiag_Block";
    bool isAutoNumber = true;
    int number = 1;
    var progLang = ProgrammingLanguage.ProDiag;
    FB block = blockComposition.CreateFB(fbName, isAutoNumber, number, progLang);
    string idbName="ProDiag_IDB";
    string instanceOfName = fbName;
    InstanceDB idbBlock = blockComposition.CreateInstanceDB(idbName, isAutoNumber, number,
instanceOfName);
}
```

Siehe auch

Auf Überwachungen und Eigenschaften des ProDiag-FB zugreifen (Seite 523)

5.11.3.13 Auf Überwachungen und Eigenschaften des ProDiag-FB zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Auf Überwachungen des User-FB zugreifen

Sie können mit TIA Portal Openness auf die Überwachungen am FB-Baustein zugreifen. Jeder FB-Baustein hat die Liste der Überwachungen einschließlich Classic-PLCs und Plus-PLCs. Sie können AssignedProDiagFB am

InstanceDB über das Attribut AssignedProDiagFB festlegen (siehe Bausteine exportieren (Seite 1395)).

Sie können mit den Methoden GetAttribute(), GetAttributes() und and SetAttribute() auf die Attribute zugreifen.

Einschränkung

Sie können mit SetAttributes() in TIA Portal Openness nicht mehrere Attribute festlegen. Beim Versuch, mit SetAttributes() mehrere Attribute festzulegen, wird in TIA Portal Openness eine Ausnahme ausgegeben.

Programmcode: Auf Überwachungen des ProDiag-FB zugreifen

```
...
PlcBlock iDB = plc.BlockGroup.Blocks.Find("FB_Block_DB");
string fbName = iDB.GetAttribute("InstanceOfName").ToString();
FB fb = (FB)plc.BlockGroup.Blocks.Find(fbName);
if (fb.Supervisions.Count > 0) Console.WriteLine("Contains supervisions");
else
    Console.WriteLine("Does not contains supervisions");
...
```

Programmcode: Zugeordneten ProDiag-FB und IDB abrufen und festlegen

```
...
PlcBlockGroup blockFolder = plc.BlockGroup;
PlcBlock instanceDB = blockFolder.Blocks.Find("IDB");
PlcBlock plcProdiag = blockFolder.Blocks.Find("block_Prodiag");
instanceDB.SetAttribute("AssignedProDiagFB", plcProdiag.Name);
var assignedProDiagFB = instanceDB.GetAttribute("AssignedProDiagFB");
...
```

Siehe auch

Einen ProDiag-FB anlegen (Seite 522)

5.11.3.14 ProDiag-FB für DBs & Variablentabelle zuweisen**Voraussetzung**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Mit TIA Portal Openness können Sie den zugewiesenen ProDiagFB-Wert eines Mitglieds im globalen DB und der Variablentabelle mit GetAttribute/SetAttribute abrufen und einstellen.

Der Typname Siemens.Engineering.SW.Blocks.Interface.Member stellt die folgenden Eigenschaften bereit:

Name der Eigenschaft	Datentyp	Zugriff
Name	String	Lesen
AssignedProDiagFB	String	Lesen/Schreiben

Programmcode: proDiagFB-Wert von DB-Mitglied & Variablen-Mitglied abrufen

Ändern Sie den folgenden Programmcode, um ProDiagFB in verschiedenen Instanzen des UDT abzurufen, unabhängig davon, ob der UDT überwacht wird oder nicht:

```
//Simple UDT instance
MemberComposition gdbMembers = dataBlock.Interface.Members;
Member member = gdbMembers.Find("SimpleUdtInstanceMember");
object proDiagFB = member.GetAttribute("AssignedProDiagFB");
//Array of UDT Instance
Member member = dataBlock.Interface.Members.Find("ArrayUdtMember.ArrayUdtMember[0]");
object proDiagFB = member.GetAttribute("AssignedProDiagFB");
//UDT instance inside Struct
Member member = dataBlock.Interface.Members.Find("structMember.UdtInstanceMember");
object proDiagFB = member.GetAttribute("AssignedProDiagFB");
//UDT Instance inside Array of Struct
MemberComposition gdbMembers = dataBlock.Interface.Members;
Member member =
gdbMembers.Find("ArrayOfStructMember.ArrayOfStructMember[0].UdtInstanceMember");
object proDiagFB = member.GetAttribute("AssignedProDiagFB");
```

Ändern Sie den folgenden Programmcode, um ProDiagFB in der Variablentabelle abzurufen:

```
// UDT Instance in TagTable
PlcTagTable tagTable = plc.TagTableGroup.TagTables.Find("Tag table_1");
PlcTag tag = tagTable.Tags.Find("Tag_1");
object proDiagFB = tag.GetAttribute("AssignedProDiagFB ");
```

Programmcode: ProDiagFB-Wert für DB & Variablen-Mitglied festlegen

Ändern Sie den folgenden Programmcode, um das zugewiesene ProDiagFB-Attribut für ein Mitglied im DB festzulegen:

```
//Simple UDT instance
MemberComposition gdbMembers = dataBlock.Interface.Members;
Member member = dataBlock.Interface.Members.Find("SimpleUdtInstanceMember");
member.SetAttribute("AssignedProDiagFB", "ProDiagFBName");
//Array of UDT Instance
Member member = dataBlock.Interface.Members.Find("ArrayUdtMember.ArrayUdtMember[0]");
member.SetAttribute("AssignedProDiagFB", "ProDiagFBName");
//UDT instance inside Struct
Member member = dataBlock.Interface.Members.Find("structMember.UdtInstanceMember");
member.SetAttribute("AssignedProDiagFB", "ProDiagFBName");
//UDT Instance inside Array of Struct
Member member =
dataBlock.Interface.Members.Find("ArrayOfStructMember.ArrayOfStructMember[0].UdtInstanceMember");
member.SetAttribute("AssignedProDiagFB", "ProDiagFBName");
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie den folgenden Programmcode, um das zugewiesene ProDiagFB-Attribut für ein Variablenmitglied festzulegen:

```
// UDT Instance in TagTable
var tagTable = plc.TagTableGroup.TagTables.Find("Tag table_1");
var tag = tagTable.Tags.Find("Tag_1");
tag.SetAttribute ("AssignedProDiagFB ", "ProDiagFBName");
```

Hinweis

GetAttribute und SetAttribute von ProDiagFB ist ein dynamisches Attribut und nur gültig für UDT-Instanzen im globalen DB und der Variablen-tabelle.

Objekte wie einfache Datentypen wie zum Beispiel Int, String usw. und abgeleitete Datentypen wie Struct, Array, alle systemdefinierten Typen, anwender- und systemspezifische Konstantentypen unterstützen dieses Attribut nicht.

Wenn Sie versuchen, das Attribut für ein ungültiges Objekt abzurufen oder einzustellen, wird eine Ausnahme erzeugt.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.3.15 ProDiag-FB-Bausteine und -Attribute lesen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können TIA Portal Openness verwenden, um die Version des ProDiag-Funktionsbausteins und weitere Attributwerte in Verbindung mit ProDiag zu lesen. Mit GetAttribute () und GetAttributes () können Sie die vorhandenen sprachspezifischen Attribute von ProDiag-FBs lesen.

Attribute

Die folgenden Attribute werden von ProDiag-FBs in Openness unterstützt:

Attribute	Typ
ProDiagVersion	Version
InitialValueAcquisition	Bool
UseCentralTimeStamp	Bool

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.3.16 ProDiag-Meldungen exportieren

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Verwendung

In TIA Portal Openness können Sie mit ExportProDIAGInfo() am CodeBlock eine Meldung für einen bestimmten ProDiag-Baustein im CSV-Format exportieren. Die entsprechende Ausgabe kann in Microsoft Excel

geöffnet werden.

Damit Sie den ProDiag-Baustein im CSV-Format exportieren können, muss die Programmiersprache eines ProDiag-FB "ProDiag" sein. Ansonsten wird eine wiederherstellbare Ausnahme mit folgender Fehlermeldung ausgelöst:

Exportdatei

Die exportierte ProDiag-Datei enthält die Kennung bestehend aus Client-Alarm-ID und Überwachungs-ID im Hexadezimalformat. Die zweite Spalte enthält die Priorität der Überwachungskategorie.

Nachfolgend finden Sie ein Beispiel für den Inhalt einer exportierten Datei.

	A	B	C	D	E	F	G	H	I	J	K
1	Identification	Priority	Alarm text	Information text	Additional text 1	Additional text 2	Additional text 3	Additional text 4	Additional text 5	Additional text 6	Additional text 7
2	1	0	1 : SubCategory1 : SubCategory2 : Message : PLC_6	Default_SupervisionFB : 2	1 %I/O 0 : globalTag : tag comment	@4%1.2i@					
3	18	0	2 : Error text : Default_SupervisionFB : 24 : LAD_DB : :		SubCategory1 : SubCategory2						
4											

Programmcode

```
private static void ExportProDiagBlock(PlcBlock plcBlock)
{
    FB functionBlock = plcBlock as FB;
    if (functionBlock != null && functionBlock.ProgrammingLanguage ==
        ProgrammingLanguage.ProDiag && functionBlock.IsConsistent)
    {
        // Define absolute directory path where the CSV files will be stored
        string directoryPath = Path.Combine(@"D:\Temp\ProDiagInfos", functionBlock.Name);
        DirectoryInfo directoryInfo = new DirectoryInfo(directoryPath);
        // Export the alarm messages of a ProDiag block in CSV format
        functionBlock.ExportProDIAGInfo(directoryInfo);
    }
}
```

5.11.3.17 Externe Datei hinzufügen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet:
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können einem PLC eine externe Datei hinzufügen. Diese externe Datei wird im Dateisystem unter dem definierten Pfad gespeichert.

Folgende Formate werden unterstützt:

- AWL
- SCL
- DB
- UDT

Hinweis

Der Zugriff auf Gruppen im Ordner „Externe Quelldateien“ wird nicht unterstützt.

Es wird eine Ausnahme ausgelöst, wenn Sie eine andere Dateierweiterung als *.AWL, *.SCL, *.DB oder *.UDT angeben.

Programmcode

Ändern Sie den folgenden Programmcode, um im Ordner "Externe Quelldateien" eine externe Datei aus einem Baustein zu erzeugen:

```
private static void CreateBlockFromFile(PlcSoftware plcSoftware)
// Creates a block from a AWL, SCL, DB or UDT file
{
    PlcExternalSource externalSource =
    plcSoftware.ExternalSourceGroup.ExternalSources.CreateFromFile("SomeBlockNameHere", "SomePathHere");
}
```

5.11.3.18 Quelle aus Baustein generieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Einleitung

Die Schnittstelle TIA Portal Openness API unterstützt die Generierung von Quellen in UTF-8 aus AWL- oder SCL-Bausteinen, Datenbausteinen und PLC-Typen (Anwenderdatentypen). Um eine Quelldatei eines Bausteins zu generieren, rufen Sie die Methode `GenerateSource` auf der Instanz `PlcExternalSourceSystemGroup` auf.

Der Umfang der generierten Quelldatei ist von der Generierungsoption dieser Funktion abhängig:

- `GenerateOptions.None`
Quelle nur aus bereitgestellten Bausteinen generieren.
- `GenerateOptions.WithDependencies`
Quelle einschließlich aller abhängigen Objekte generieren.

Die Schnittstelle `Siemens.Engineering.SW.ExternalSources.IGenerateSource` zeigt an, dass eine Quelle generiert werden kann.

Für Bausteine werden nur die Programmiersprachen AWL und SCL unterstützt. In den folgenden Fällen werden Ausnahmen ausgelöst:

- Die Programmiersprache ist nicht AWL oder SCL
- Eine Datei gleichen Namens ist bereits am Zielspeicherort vorhanden.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Für Anwenderdatentypen wird nur die Dateierweiterung "*.udt" unterstützt. In den folgenden Fällen werden Ausnahmen ausgelöst:

- Die Dateierweiterung für DBs ist nicht "*.db".
- Die Dateierweiterung für AWL-Bausteine ist nicht "*.awl".
- Die Dateierweiterung für SCL-Bausteine ist nicht "*.scl".

Programmcode

Um Quelldateien aus Bausteinen und Typen zu erzeugen, ändern Sie folgenden Programmcode:

```
PlcExternalSourceSystemGroup.GenerateSource(IEnumerable<IGenerateSource> plcBlocks,
FileInfo sourceFile, GenerateOptions generateOptions);
// ...
//examples
// ...
var blocks = new List<PlcBlock>(){block1};
var fileInfo = new FileInfo(@"C:\temp\SomePathHere.scl");
PlcExternalSourceSystemGroup systemGroup = ...;
systemGroup.GenerateSource(blocks, fileInfo, GenerateOptions.WithDependencies);
// exports all blocks and with all their dependencies(e.g. called blocks, used DBs or UDTs)
// as ASCII text into the provided source file.
// ...
// or
// ...
var types = new List<PlcType>(){udt1};
var fileInfo = new FileInfo(@"C:\temp\SomePathHere.udt");
PlcExternalSourceSystemGroup systemGroup = ...;
systemGroup.GenerateSource(types, fileInfo, GenerateOptions.WithDependencies);
// exports all data types and their used data types into the provided source file.
// ...
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.11.3.19 Bausteine aus Quelle erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Verwendung

In der Gruppe "Externe Quelldateien" können Sie Bausteine aus allen externen Dateien erzeugen. Es werden nur externe Dateien mit dem Format ASCII unterstützt.

Hinweis

Vorhandene Bausteine werden überschrieben.

Wenn beim Aufruf ein Fehler auftritt, wird eine Exception ausgelöst. Vor TIA Portal Openness V18 sind die ersten 256 Zeichen jeder Fehlermeldung in der Benachrichtigung über eine Exception enthalten. Ab TIA Portal Openness V18 werden die ersten 4096 Zeichen der Nachricht zurückgegeben.

Das Projekt wird auf den Verarbeitungszustand vor der Ausführung der Methode `GenerateBlocksFromSource` zurückgesetzt.

Programmcode

Um in der Gruppe "Externe Quelldateien" aus allen externen Dateien Bausteine zu erzeugen, ändern Sie folgenden Programmcode:

```
// Creates a block from an external source file
PlcSoftware plcSoftware = ...;
foreach (PlcExternalSource plcExternalSource in
plcSoftware.ExternalSourceGroup.ExternalSources)
{
    plcExternalSource.GenerateBlocksFromSource();
}
```

5.11.3.20 Generierung aus einer Quelle mit bekanntem Quellformat

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Der PLC ist nicht online.

Anwendung

Die TIA Portal Openness API-Schnittstelle unterstützt die Generierung von Baustein- und UDT-Objekten. Unter bestimmten Bedingungen schlägt die Übersetzung generierter Bausteine und UDT fehl.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Typische fehlende oder ungültige Teile, die dazu führen, dass der Baustein oder UDT nicht übersetzbar ist:

- Verweis auf einen nicht vorhandenen Typ
- Verwendung eines fehlenden Symbols

Die Generierung von UDT und Bausteinen mit `GenerateBlocksFromSource()` schlägt mit folgendem Ergebnis fehl:

- Eine wiederherstellbare Ausnahme wird ausgelöst
- Die Fehlermeldung(en) aus der Generierung der Ausnahme-Meldung
- Der generierte Baustein/UDT wird gelöscht

Eine neue Überladung des Aufrufs von `GenerateBlocksFromSource()` mit `GenerateBlockOptions.KeepOnError` führt zu folgendem Ergebnis:

- Unabhängig von den Generierungsergebnissen wird keine wiederherstellbare Ausnahme zurückgegeben
- Keine Informationen über die Generierungsergebnisse der erzeugten Bausteine
- Eine Liste der `IEngineeringObjects` der generierten Bausteine (mit oder ohne Generierungsfehler) wird zurückgegeben

Wenn der Aufruf von `GenerateBlocksFromSource()` mit `GenerateBlockOptions.None` und erfolgt, bleibt die Generierung von Baustein und UDT fehlerfrei und hat folgendes Ergebnis:

- Eine wiederherstellbare Ausnahme wird bei fehlgeschlagener Generierung ausgelöst
- Die Fehlermeldung(en) der Generierung wird (werden) in die Meldung der Ausnahme einbezogen
- Der generierte Baustein/UDT wird gelöscht

Bei fehlerfreier Generierung

- Wird keine Ausnahme zurückgegeben
- Werden keine Informationen über die Generierungsergebnisse der erzeugten Bausteine geliefert
- Wird eine `ICollection` der `IEngineeringObjects` der generierten Bausteine zurückgegeben
 - Dies ist das gleiche Verhalten wie beim Aufruf mit `GenerateBlockOptions.KeepOnError`.

Programmcode

```

try
{
    IList<IEngineeringObject> generatedObjects =
externalSource.GenerateBlocksFromSource(GenerateBlockOptions.KeepOnError);
    foreach (IEngineeringObject engineeringObject in generatedObjects)
    {
        CompilerResult compilerResult = null;
        string objectName = null;
        if (engineeringObject is PlcBlock)
        {
            // handle case for PlcBlock
            // e.g. retrieve the compiler result
            PlcBlock block = (PlcBlock)engineeringObject;
            objectName = block.Name;
            compilerResult = block.Compile();
        }
        else if (engineeringObject is PlcType)
        {
            // handle case for PlcType
            // e.g. retrieve the compiler result
            PlcType plcType = (PlcType)engineeringObject;
            objectName = plcType.Name;
            compilerResult = plcType.Compile();
        }
        // handle the compiler result
        if (compilerResult != null)
        {
            if (compilerResult.State == CompilerResultState.Error)
            {
                Console.WriteLine("Object '{0}' could not be compiled successfully!", objectName);
                Console.WriteLine("Number of compiler errors: {0}", compilerResult.ErrorCount);
                foreach (CompilerResultMessage compilerResultMessage in compilerResult.Messages)
                {
                    Console.WriteLine(compilerResultMessage.Description);
                }
            }
            else
            {
                Console.WriteLine("Object '{0}' could be compiled successfully.", objectName);
            }
        }
    }
}
catch (RecoverableException exception)
{
    // handle recoverable exception
    Console.WriteLine(exception.Message);
}

```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.3.21 Anwenderdatentyp löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Programmcode

Ändern Sie den folgenden Programmcode, um einen Benutzertyp zu löschen:

```
private static void DeleteUserDataTypes(PlcSoftware plcSoftware)
{
    PlcTypeSystemGroup typeGroup = plcSoftware.TypeGroup;
    PlcTypeComposition dataTypes = typeGroup.Types;
    PlcType dataType = dataTypes.Find("DataTypeName");
    if (dataType != null)
    {
        dataType.Delete();
    }
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.11.3.22 Auf Datentypen des PLC-Systems zugreifen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Mit Hilfe der Eigenschaften `Siemens.Engineering.SW.PlcSystemTypeGroup` und `Siemens.Engineering.SW.PlcSystemTypeGroupComposition` können Sie auf eine Liste von systemgenerierten Typen zugreifen.

Die `PlcSystemTypeGroupComposition` gibt Zugriff auf Typgruppen des PLC-Systems, während die `PlcSystemTypeGroup` den Zugriff auf die Datentypen des jeweiligen Systems ermöglicht.

Programmcode

Ändern Sie den folgenden Programmcode, um auf die Datentypen des Systems zuzugreifen:

```
private static void accessPlcSystemTypeGroup(PlcSystemTypeGroupComposition, m_target)
{
    // SystemTypeGroups is a composition of system data types
    PlcSystemTypeGroup plcSystemTypeGroup =
    m_target.TypeGroup.SystemTypeGroups.FirstOrDefault();
}
```

Siehe auch

Projekt öffnen (Seite 131)

5.11.3.23 Externe Datei löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet:
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Programmcode

Um in der Gruppe "Externe Quelldateien" eine externe Datei zu löschen, ändern Sie folgenden Programmcode:

Hinweis

Der Zugriff auf Gruppen in der Gruppe "Externe Quelldateien" wird nicht unterstützt.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

```
// Deletes an external source file
private static void DeleteExternalSource(PlcSoftware plcSoftware)
{
    PlcExternalSource externalSource =
    plcSoftware.ExternalSourceGroup.ExternalSources.Find("myExternalsource");
    externalSource.Delete();
}
```

5.11.3.24 Bausteineditor starten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine Instanz des TIA Portals ist mit Benutzeroberfläche geöffnet.

Programmcode

Um den zugehörigen Editor für eine Objektreferenz des Typs `PlcBlock` in der Instanz des TIA Portals zu starten, ändern Sie den folgenden Programmcode:

```
//Opens a block in a block editor
private static void StartBlockEditor(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    plcBlock.ShowInEditor();
}
```

Ändern Sie folgenden Programmcode, um den zugehörigen Editor für eine Objektreferenz des Typs `PlcType` in der Instanz des TIA Portals zu öffnen:

```
//Opens a udt in udt editor
private static void StartPlcTypeEditor(PlcSoftware plcSoftware)
{
    PlcTypeComposition types = plcSoftware.TypeGroup.Types;
    PlcType udt = types.Find("my_udt");
    udt.ShowInEditor();
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.11.3.25 Bausteine mit Hilfe von Fingerabdrücken ändern

Voraussetzung

- Die Anwendung ist über TIA Portal Openness mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

TIA Portal Openness unterstützt Fingerabdrücke mit dem Namen "Program Code". Dieser vergleicht den reinen Programmcode und ignoriert mehrzeilige Kommentare, Zeilenumbrüche und Tabulatoren. Die FingerprintId "ProgramCode" in Openness dient dazu, den Prüfsummenwert des Fingerabdrucks "Program Code" zu extrahieren.

Über TIA Portal Openness können Sie Änderungen innerhalb von Bausteinen und UDTs erkennen. Bei dieser Funktion können Sie die Fingerabdrücke des Objekts miteinander vergleichen. Eine Fingerabdruck-Instanz enthält eine FingerprintId, die den Typ des Fingerabdrucks definiert, sowie den Fingerabdruck-Wert als String.

Alle bereitgestellten fingerprints berücksichtigen nur Benutzereingaben, kein Übersetzungsergebnis oder andere Änderungen, die das System vornimmt. Sie können den FingerprintProvider verwenden, um Instanzen von Fingerabdrücken aufzurufen. Der FingerprintProvider-Dienst ist für Bausteine (FB, FC, OB, DB) und UDTs, jedoch nicht für Variablentabellen verfügbar. GetFingerprints() wird für eine Instanz von Fingerabdrücken aufgerufen, um alle verfügbaren Fingerabdruck-Objekte zu berechnen und zurückzugeben. Um gültige Fingerabdrücke zu erhalten, muss der Baustein oder UDT vor dem Aufrufen von Fingerabdrücken konsistent sein. Andernfalls wird eine RecoverableException ausgelöst. Ist ein Fingerabdruck auch nach seiner Berechnung noch ungültig, wird eine RecoverableException ausgegeben.

Die Enumeration FingerprintId listet alle Arten von Fingerabdrücken auf, die in Openness unterstützt werden:

Wert	Beschreibung
Code	Berücksichtigt alle Änderungen am Code im Hauptteil des Bausteins. Das Übersetzungsergebnis wird nicht berücksichtigt.
Interface	Berücksichtigt alle Änderungen an der Schnittstelle eines Bausteins, einschließlich der Startwerte eines DB.
Properties	Berücksichtigt Änderungen an den Eigenschaften eines Bausteins, z. B. Name, Nummer.
Comments	Berücksichtigt Änderungen an den Kommentaren eines Bausteins. Bei OBs ändert sich der Fingerabdruck auch, wenn die Liste verfügbarer Sprachen in den Spracheinstellungen des Projekts geändert wird.
LibraryType	Vorhanden, wenn ein Baustein mit einem Bibliothekstyp verschaltet ist.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Wert	Beschreibung
Texts	Bei V15 SP1 ist dieser Fingerabdruck nur für Graph-Bausteine vorhanden.
Alarms	Vorhanden, wenn ein Baustein mit der Meldungsfunktion arbeitet.
Supervision	Vorhanden, wenn ein Baustein eine Überwachungsfunktion enthält.
TechnologyObject	Nur für Technologieobjekt-DBs vorhanden.
Events	Nur für OBs vorhanden.
TextualInterface	Vorhanden, wenn der Baustein eine Textschnittstelle besitzt.
ProgramCode	Berücksichtigt ausschließlich Änderungen von Programmcode und ignoriert mehrzeilige Kommentare, Zeilenumbrüche und Tabulatoren

Programmcode

Ändert den folgenden Programmcode, um den Prüfsummenwert des Fingerabdrucks "Program Code" zu extrahieren:

```
var blocks = plcSoftware.BlockGroup.Blocks;
var block1 = blocks.Find("FBDBlock") as CodeBlock;
FingerprintProvider provider = block1.GetService<FingerprintProvider>();
IList<Fingerprint> fingerprints = provider.GetFingerprints();
foreach(var fingerprint in fingerprints)
{
    string fpValue = fingerprint.Value;
    FingerprintId fpId = fingerprint.Id;
    Console.WriteLine("Fingerprint Id:" + fingerprint.Id + " checksum :"+ fingerprint.Value);
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.3.26 Bausteine für benutzerdefinierte Seiten generieren/löschen**Voraussetzung**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Sie können mit TIA Portal Openness benutzerdefinierte Seiten innerhalb des Webservers von PLCs erstellen: Die Daten für diese benutzerdefinierten Seiten werden in speziellen, systemgenerierten Bausteinen der PLC gespeichert. Sie können diese Bausteine auch manuell bearbeiten und löschen. Wenn Sie jedoch Daten oder Eigenschaften in diesen Datenbausteinen ändern, funktioniert der Webserver anschließend nicht mehr ordnungsgemäß.

In TIA Portal Openness wird der neue Dienst `WebserverUserDefinedPages` an der PLC des `DeviceItem` hinzugefügt. Sie können mit Hilfe dieses Dienstes die Datenbausteine für benutzerdefinierte Seiten generieren: `List<PLCBlock> GenerateBlocks(Arguments)`. Beim Aufruf dieser Funktion werden die entsprechenden Datenbausteine generiert und eine Liste der generierten Datenbausteine wird an den Benutzer zurückgegeben.

Es gibt zwei mögliche Überladungen dieser Funktion:

- `IList<PlcBlock> WebserverUserDefinedPages.GenerateBlocks(WebDBGenerateOptions GenerateOptions);`
- `IList<PlcBlock> WebserverUserDefinedPages.GenerateBlocks(System.IO.DirectoryInfo htmlDirectory, System.IO.FileInfo defaultHTMLPage, string applicationName, WebDBGenerateOptions GenerateOptions);`

Die Parameterwerte für das HTML-Verzeichnis, die HTML-Seite und den Anwendungsnamen werden nicht in den Projektdaten gespeichert. Somit werden die Werte der entsprechenden dynamischen Attribute nicht geändert.

Parameter	Datentyp	Typ	Beschreibung
GenerateOptions	WebDBGenerateOptions (enumeration)	Mandatory	Mögliche Werte: • None - Wenn Datenbausteine für benutzerdefinierte Seiten bereits vorher erstellt wurden, wird keine Aktion ausgeführt und eine Ausnahme ausgelöst • Override - Wenn Datenbausteine für benutzerdefinierte Seiten bereits vorher erstellt wurden, werden sie gelöscht und die neuen Datenbausteine generiert
htmlDirectory	System.IO.DirectoryInfo	Optional	Gibt das HTML-Verzeichnis unabhängig vom Wert des dynamischen Attributs <code>WebserverHTMLDirectory</code> an.
defaultHTMLPage	System.IO.FileInfo	Optional	Gibt die standardmäßige HTML-Seite unabhängig vom Wert des dynamischen Attributs <code>WebserverDefaultHTMLPage</code> an.
applicationName	string	Optional	Gibt den Anwendungsnamen unabhängig vom Wert des dynamischen Attributs <code>WebserverApplicationName</code> an.

GenerateBlocks() löst eine wiederherstellbare Ausnahme aus:

- Wenn der Webserver deaktiviert ist (WebserverActive == False) - "Webserver muss aktiviert werden"
- Wenn WebserverHTMLDirectory leer oder der Pfad ungültig ist - "HTML-Verzeichnis hat ungültigen Pfad"
- Wenn WebserverdefaultHTMLPage leer oder der Pfad ungültig ist - "Default-HTML-Verzeichnis hat ungültigen Pfad"
- Wenn das WebserverHTMLDirectory zu lang ist - "HTML-Verzeichnis hat zu viele Daten"
- Der Anwendungsname ist ungültig - "Der Applikationsname ist ungültig"
- Die dynamischen Inhalte sind ungültig - "Dynamischer Inhalt ist ungültig"
- Die Nummer des Steuerdatenbausteins ist ungültig - "Die Nummer des Steuer-DBs ist ungültig"
- Die Startnummer des Datenbausteins ist ungültig - "Die Startnummer des DBs ist ungültig"
- Bausteine sind bereits vorhanden: Wenn der Benutzer versucht, einen Baustein mit einem Namen zu generieren, der bereits vorhanden ist - "Löschen Sie die bestehenden Bausteine, bevor Sie neue Bausteine generieren" (wird nur ausgelöst, wenn WebDBGenerateOptions.None verwendet wurde)

Es gibt keine spezielle Funktion zum Löschen aller Bausteine, die mit benutzerdefinierten Seiten verbunden sind. Sie müssen manuell die Bausteine löschen, die von der Aktion "Generieren" an die Seiten zurück übergeben werden, oder zu den entsprechenden Datenbausteinen navigieren, wie es in TIA Portal Openness bereits möglich ist.

Programmcode

Ändern Sie den folgenden Programmcode, um einen Baustein für eine benutzerdefinierte Seite zu generieren:

```
DirectoryInfo htmlDirectory = ...;
FileInfo defaultHTMLPage = ...;
string applicationName = ...;
DeviceItem deviceItem = ...;
var WebserverUserDefinedPagesService = deviceItem.GetService<WebserverUserDefinedPages>();
IList<PlcBlock> block =
WebserverUserDefinedPagesService.GenerateBlocks(WebDBGenerateOptions.None);
IList<PlcBlock> blocks = WebserverUserDefinedPagesService.GenerateBlocks(htmlDirectory,
defaultHTMLPage, applicationName, WebDBGenerateOptions.Override);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.3.27 Schreibzugriff für Prioritätsattribut des OB-Bausteins

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können TIA Portal Openness dafür verwenden, Schreibzugriff für Bausteineigenschaften zu gewähren, insbesondere für OBs. Sie können die Priorität eines Bausteins sowohl im Programmiermodell unit als auch im Programmiermodell non-unit mit Hilfe von TIA Portal Openness ändern.

Die folgende Eigenschaft ist für den Zugriff auf einen Baustein verfügbar:

Name der Eigenschaft	Datentyp	Zugriff
PriorityNumber	Integer	Lesen/Schreiben

Programmcode

Um die Priorität für den OB-Baustein einzustellen, ändern Sie folgenden Programmcode:

```
PlcSoftware plcSoftware = ...;
PlcBlock obBlock = plcSoftware.BlockGroup.Blocks.Find("CyclicInterrupt");
obBlock.SetAttribute("PriorityNumber", 14);
```

Siehe auch

Projekt öffnen (Seite 131)

5.11.3.28 Bausteine/UDTs aus einer externen Quelldatei in einer bestimmten Benutzergruppe erzeugen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können mit TIA Portal Openness Bausteine/UDTs unter bestimmten Benutzergruppen sowohl in der PLC-Programmierung als auch in der Einheiten-Programmierung erzeugen.

Funktionen

Name der Methode mit Signatur

- `Public IList<IEngineeringObject>
PlcExternalSource.GenerateBlocksFromSource(PlcBlockUserGroup blockuserGroup,
GenerateBlockOption generateBlockOption)`
- `Public IList<IEngineeringObject>
PlcExternalSource.GenerateBlocksFromSource(PlcBlockUserGroup blockuserGroup,
GenerateBlockOption generateBlockOption)`

Programmcode: Generieren von Bausteinen aus einer Quelle unter bestimmten Benutzergruppen

Ändern Sie folgenden Programmcode, um die Informationen zur Benutzergruppe, unter der die Bausteine erzeugt werden sollen, abzurufen:

```
Siemens.Engineering.SW.Blocks.PlcBlockUserGroup folder =  
plc.BlockGroup.Groups.Find("Group_1");
```

Ändern Sie folgenden Programmcode, um Bausteine unter Gruppen ohne Optionen zu erzeugen:

```
plc.ExternalSourceGroup.ExternalSources.Find("Block_1.scl").GenerateBlocksFromSource(folder,  
GenerateBlockOption.None);
```

Ändern Sie folgenden Programmcode, um Bausteine mit Optionen als KeepOnError zu erzeugen:

```
plc.ExternalSourceGroup.ExternalSources.Find("Block_1.scl")  
GenerateBlocksFromSource(folder, GenerateBlockOption.KeepOnError);
```

Programmcode: Erzeugen von UDTs aus einer Quelle unter bestimmten Benutzergruppen

```
Siemens.Engineering.SW.Types.PlcTypeUserGroup folder1 =  
plc.TypeGroup.Groups.Find("Group_1");
```

Ändern Sie folgenden Programmcode, um UDTs unter Gruppen ohne Optionen zu erzeugen:

```
plc.ExternalSourceGroup.ExternalSources.Find("User_data_type_1.udt").GenerateBlocksFromSource(folder1, GenerateBlockOption.None);
```

Ändern Sie folgenden Programmcode, um Bausteine mit Optionen als KeepOnError zu erzeugen:

```
plc.ExternalSourceGroup.ExternalSources.Find("User_data_type_1.udt").GenerateBlocksFromSource(folder1, GenerateBlockOption.KeepOnError);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.3.29 Ladbare Datei für Bausteine außerhalb von Software Units erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Das TIA Portal-Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC muss sauber übersetzt sein

Einleitung

Sie können mit TIA Portal Openness über die folgende Methode eine ladbare Datei für Bausteine außerhalb von Software Units erzeugen: `GenerateLoadable(FileInfo path, IEnumerable<PlcBlock> blocks, TargetOption targetOption)`

Hinweis

Das Erzeugen einer ladbaren Datei für Bausteine wird bei der PLC1518 oder einem Untertyp und PLCs mit einer Firmware V2.8 oder höher unterstützt.

Einschränkung

Bei der Auswahl des Bausteins gelten folgende Einschränkungen:

- Der Baustein darf kein Systembaustein sein
- Der Baustein darf kein PRODIAG-Baustein sein
- Der Baustein darf kein Baustein mit Fehlersicherheit sein

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

- Wenn der Baustein ein OB ist, darf er nur vom Typ ProgramCycle sein
- Der Baustein darf keine dynamische Bindung aufweisen
- Der Baustein darf kein Technologieobjekt (TO) sein

Parameter und Enum-Werte

GenerateLoadable() akzeptiert zur Erzeugung einer ladbaren Datei für Bausteine folgende Parameter:

Parametername	Datentyp	Beschreibung	Hinweis
path	FileInfo	Gibt Namen und Speicherort der ladbaren Datei an.	Bereits vorhandene ladbare Dateien können nicht ersetzt werden und das Erstellen des übergeordneten Verzeichnisses ist nicht möglich.
blocks	System.Collections.Generic.IEnumerable<Siemens.Engineering.SW.Blocks.PlcBlock>	Gibt die Liste der Programmbausteine außerhalb von Software Units zur Erzeugung der ladbaren Datei an	
targetOption	Siemens.Engineering.SW.Loader.TargetOption	Gibt an, ob die ladbare Datei für eine echte oder simulierte PLC erstellt werden soll.	Wenn TargetOption.PlcSim übergeben wird, müssen alle Bausteine mit Simulationsunterstützung übersetzt worden sein

Der Parameter TargetOption verwendet zur Erzeugung einer ladbaren Datei folgende Enum-Werte:

Wert	Beschreibung
TargetOption.None	Der Standardwert sollte nicht verwendet werden.
TargetOption.Plc	Zur Erzeugung einer ladbaren Datei für eine echte PLC.
TargetOption.PlcSim	Zur Erzeugung einer ladbaren Datei für eine simulierte PLC.

Programmcode

Ändern Sie folgenden Programmcode, um eine ladbare Datei für Bausteine außerhalb von Software Units zu erzeugen:

```
private void GenerateLoadableProgramBlock(string loadablePath)
{
    // Getting an access to the PLC device of the project
    PlcSoftware plc = GetPlc();
    // The GetService() will return null if the PLC is not supported.
    var loadableProviderService = plc.GetService<LoadableProvider>();
    // Create a variable file with name "file" to access the loadable path
    var file = new FileInfo(loadablePath);
    // Create a variable with name "blocks" for a list of program blocks
    // from the non-software unit program.
    var blocks = new List<PlcBlock>
    {
        // Accessing a specific block
        // with name "Block_1" under plc
        plc.BlockGroup.Blocks.Find("Block_1") ?? throw new InvalidOperationException()
    };
    // The loadableProviderService generates a loadable file for program block
    // by calling the GenerateLoadable() with respective parameters.
    try
    {
        loadableProviderService.GenerateLoadable(file, blocks, TargetOption.Plc);
    }
    catch (EngineeringTargetInvocationException e)
    {
        // An EngineeringTargetInvocation exception is thrown
        // if the generation of the loadable file fails.
        // The exception message contains hints on
        // what caused the loadable file creation to fail.
        Console.WriteLine("Generating the loadable file failed.");
        Console.WriteLine(e);
    }
}
```

Ausnahme

Die folgenden möglichen Ausnahmemeldungen sind unten aufgeführt:

- Eine ladbare Datei kann nur aus einer ganzen Software Unit, nicht aus einem einzelnen Baustein innerhalb einer Unit erstellt werden.
- Die ausgewählten Bausteine stammen aus verschiedenen PLCs.
- Zielverzeichnis nicht gefunden: '{0}'.
- Es ist nicht möglich, eine ladbare Datei aus einem fehlersicheren Baustein zu erstellen: '{0}'.
- Zieldatei ist bereits vorhanden: '{0}'.
- Auf die ladbare Datei kann nicht zugegriffen werden.
- Ein interner Fehler ist aufgetreten.
- Die PLC muss übersetzt werden.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

- Interne Serialisierung der ladbaren Datei fehlgeschlagen.
- Die interne Serialisierung ist fehlgeschlagen, weil die ladbare Datei entfernt wurde.
- Der angegebene Pfad oder Dateiname oder beides überschreitet die vom System definierte maximale Länge.
- Die PLC '{0}' wird nicht unterstützt.
- Die übergeordnete PLC des Diensts und des Ziels weichen voneinander ab.
- Ein(ige) Baustein(e) kann (können) nicht simuliert werden. Wenn es sich bei dem Baustein um einen Bibliotheksbaustein handelt, verwenden Sie eine Bibliothek mit Simulationsunterstützung. Andernfalls wählen Sie in den Projekteigenschaften die Option "Beim Übersetzen von Bausteinen Simulierbarkeit unterstützen" ("Support simulation during block compilation") und übersetzen Sie den Baustein erneut.
- Es ist nicht möglich, eine ladbare Datei aus einem Systembaustein zu erstellen: '{0}'.
- Der Zugriff auf die Datei wird verweigert.
- Software Units, die PLC-Datentypen enthalten, werden nicht unterstützt.
- Es ist nicht möglich, eine ladbare Datei aus (einem) ausgewählten Objekt(en) zu erstellen.
- Es ist nicht möglich, eine ladbare Datei aus dem OB '{0}' zu erstellen, weil er nicht die Ereignisklasse 'Programmzyklus' aufweist.
- Es ist nicht möglich, eine ladbare Datei aus einem ProDiag-Baustein zu erstellen: '{0}'.
- Es ist nicht möglich, eine ladbare Datei aus dem ausgewählten Objekt zu erstellen: '{0}'.
- Es ist nicht möglich, eine ladbare Datei aus dem Baustein '{0}' mit aktivierter dynamischer Bindung zu erstellen.

Siehe auch

Ladbare Datei für Software Units erzeugen (Seite 647)

5.11.4 Technologieobjekte**5.11.4.1 Übersicht über Funktionen für Technologieobjekte**

TIA Portal Openness unterstützt eine Auswahl von Technologieobjektfunktionen für definierte Aufgaben, die Sie mittels der Public API außerhalb des TIA Portal aufrufen können.

In den folgenden Kapiteln finden Sie Codebeispiele, die an Ihr Openness-Programm angepasst werden können.

Funktionen

Die folgenden Funktionen sind für Technologieobjekte verfügbar:

- Zusammensetzung von Technologieobjekten abfragen (Seite 550)
- Technologieobjekt erstellen (Seite 550)
- Technologieobjekt löschen (Seite 551)
- Gruppe für Technologieobjekte erstellen (Seite 552)
- Gruppe für Technologieobjekte löschen (Seite 553)
- Technologieobjekt übersetzen (Seite 553)
- Technologieobjekte enumerieren (Seite 555)
- Gruppen für Technologieobjekte enumerieren (Seite 555)
- Technologieobjekt finden (Seite 557)
- Parameter eines Technologieobjekts enumerieren (Seite 557)
- Parameter eines Technologieobjekts finden (Seite 558)
- Parameter eines Technologieobjekts lesen (Seite 559)
- Parameter eines Technologieobjekts schreiben (Seite 560)
- Technologieobjekte exportieren (Seite 1453)
- Technologieobjekte importieren (Seite 1455)
- Mit Hardwarekomponenten verschalten (Seite 569)

Siehe auch

Standard-Bibliotheken (Seite 76)

Einsatzmöglichkeiten (Seite 52)

TIA Portal Openness-Objektmodell (Seite 55)

5.11.4.2 Übersicht über Technologieobjekte und Versionen

Technologieobjekte

Die folgende Tabelle zeigt die in der Public API verfügbaren Technologieobjekte.

CPU	Technologie	Technologieobjekt	Version des Technologieobjekts	CPU FW
S7-1200	Motion Control	TO_PositioningAxis	≥ V6.0	≥ V4.2
		TO_CommandTable		
	PID-Regelung	PID_Compact	≥ V2.3	≥ V4.2
		PID_3Step	V2.3	
		PID_Temp	V1.1	

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

CPU	Technologie	Technologieobjekt	Version des Technologieobjekts	CPU FW	
S7-1500	Motion Control	TO_SpeedAxis	≥ V3.0	≥ V2.0	
		TO_PositioningAxis			
		TO_ExternalEncoder			
		TO_SynchronousAxis			
		TO_OutputCam			
		TO_CamTrack			
		TO_MeasuringInput			
		TO_Cam (S7-1500T) ¹⁾			
		TO_Kinematics (S7-1500T)	≥ V4.0	≥ V2.6	
		TO_LeadingAxisProxy (S7-1500T)	≥ V5.0	≥ V2.8	
		TO_Cam_10k (S7-1500T) ¹⁾	≥ V6.0	≥ V2.9	
		TO_Interpreter (S7-1500T)	≥ V8.0	≥ V3.1	
		TO_InterpreterMapping (S7-1500T) ²⁾			
		TO_InterpreterProgram (S7-1500T) ²⁾			
	Zählen und Messen	High_Speed_Counter	≥ V4.1	Jede	
		SSI_Absolute_Encoder	≥ V3.1		
	PID-Regelung	PID-Regelung	PID_Compact	≥ V2.3	≥ V2.0
			PID_3Step	V2.3	
			PID_Temp	V1.1	
			CONT_C	V1.1	Jede
			CONT_S		
			TCONT_CP		
			TCONT_S		
S7-300/400			PID-Regelung	CONT_C	V1.1
	CONT_S				
	TCONT_CP				
	TCONT_S				
	TUN_EC ²⁾				
	TUN_ES ²⁾				
	PID_CP ²⁾	V2.0			
	PID_ES ²⁾				
	EMC	AXIS_REF	V2.0		

¹⁾ Das Technologieobjekt unterstützt die folgenden Openness-Funktionen nicht: Schreiben von Parametern.

²⁾ Das Technologieobjekt unterstützt die folgenden Openness-Funktionen nicht: Enumerieren von Parametern, Suchen von Parametern, Lesen von Parametern, Schreiben von Parametern.

Siehe auch

S7-1500 Motion Control (Seite 569)

5.11.4.3 Überblick der Datentypen

Die Datentypen der Parameter von Technologieobjekten im TIA Portal sind C#-Datentypen in der Public API zugeordnet.

Datentypen

Die folgende Tabelle zeigt die Zuordnung der Datentypen:

Format	Datentyp im TIA Portal	Datentyp in C#
Binärzahlen	Bool	bool
	BBool	bool
	Byte	byte
	Word	ushort
	DWord	uint
	LWord	ulong
Ganzzahlen	SInt	sbyte
	Int	short
	Dint	int
	Lint	long
	USInt	byte
	UInt	ushort
	UDint	uint
	ULInt	ulong
Gleitpunktzahlen	Real	float
	LReal	double
	Time	double
Zeichenfolgen	Char	char
	WChar	char
	String	string
	WString	string
Hardware-Datentypen	HW_*	ushort
	Block_*	ushort

* Platzhalter für Erweiterung des Gerätetyps im TIA Portal-Projekt

5.11.4.4 Zusammensetzung von Technologieobjekten abfragen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe [Verbindung zum TIA Portal aufbauen](#) (Seite 84)
- Ein Projekt ist geöffnet.
Siehe [Projekt öffnen](#) (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe [PLC-Target und HMI-Target abfragen](#) (Seite 277)

Programmcode

Um alle Technologieobjekte einer PLC zu erhalten, ändern Sie den folgenden Programmcode:

```
// Retrieves all technology objects of a PLC
private static void GetTechnologicalObjectsOfPLC(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBGGroup technologicalObjectGroup =
    plcSoftware.TechnologicalObjectGroup;
    TechnologicalInstanceDBComposition technologicalObjects =
    technologicalObjectGroup.TechnologicalObjects;
}
```

Siehe auch

[Standard-Bibliotheken](#) (Seite 76)

5.11.4.5 Technologieobjekt erstellen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe [Verbindung zum TIA Portal aufbauen](#) (Seite 84)
- Ein Projekt ist geöffnet.
Siehe [Projekt öffnen](#) (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe [PLC-Target und HMI-Target abfragen](#) (Seite 277)

Verwendung

Nur Technologieobjekte, die im Abschnitt [Übersicht über Technologieobjekte und Versionen](#) (Seite 547) aufgeführt sind, können erstellt werden. Bei nicht unterstützten Technologieobjekten oder ungültigen Parametern wird eine Ausnahme gemeldet. Siehe auch [Umgang mit Exceptions](#) (Seite 1229).

Programmcode

Um ein Technologieobjekt zu erstellen und es einer vorhandenen PLC hinzuzufügen, ändern Sie den folgenden Programmcode:

```
// Create a technology object and add to technology object composition
private static void CreateTechnologicalObject(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBComposition technologicalObjects =
    plcSoftware.TechnologicalObjectGroup.TechnologicalObjects;

    string nameOfTO = "PID_Compact_1"; // How the technology object should be named
    string typeOfTO = "PID_Compact"; // How the technology object type is called, e.g. in
    // "Add new technology object"-dialog
    Version versionOfTO = new Version("2.3"); // Version of technology object
    TechnologicalInstanceDB technologicalObject = technologicalObjects.Create(nameOfTO,
    typeOfTO, versionOfTO);
}
```

Mögliche Werte und Kombinationen aus Name, Typ und Version des Technologieobjekts sind im Abschnitt Übersicht über Technologieobjekte und Versionen (Seite 547) zu finden.

Siehe auch

Standard-Bibliotheken (Seite 76)
S7-1500 Motion Control (Seite 569)

5.11.4.6 Technologieobjekt löschen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)
- Das Technologieobjekt ist vorhanden.
Siehe Technologieobjekt finden (Seite 557)

Programmcode

Um ein Technologieobjekt zu löschen, ändern Sie den folgenden Programmcode:

```
// Delete a technology object from DB composition and from PLC
private static void DeleteTechnologicalObject(TechnologicalInstanceDB technologicalObject)
{
    technologicalObject.Delete();
}
```

Siehe auch

Standard-Bibliotheken (Seite 76)

5.11.4.7 Gruppe für Technologieobjekte erstellen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe [Verbindung zum TIA Portal aufbauen](#) (Seite 84)
- Ein Projekt ist geöffnet.
Siehe [Projekt öffnen](#) (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe [PLC-Target und HMI-Target abfragen](#) (Seite 277)

Programmcode

Um eine Gruppe für Technologieobjekte zu erstellen, ändern Sie folgenden Programmcode:

```
private static void CreateTechnologicalObjectGroup(PlcSoftware plcsoftware)
//Create a user group for technology objects
{
    TechnologicalInstanceDBGGroup systemGroup = plcsoftware.TechnologicalObjectGroup;
    TechnologicalInstanceDBUserGroupComposition groups = systemGroup.Groups;
    TechnologicalInstanceDBUserGroup myGroup = groups.Create("MySubGroupName");
}
```

5.11.4.8 Gruppe für Technologieobjekte löschen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)

Programmcode

Um eine Gruppe für Technologieobjekte zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteTechnologicalObjectGroup(PlcSoftware plcsoftware)
// Delete groups for technology objects
{
    TechnologicalInstanceDBGroup group
        = plcsoftware.TechnologicalObjectGroup.Groups.Find("myGroup");
    TechnologicalInstanceDBUserGroupComposition subgroups = group.Groups;
    TechnologicalInstanceDBUserGroupComposition subgroup = subgroups.Find("myUserGroup");
    if (subgroup != null)
    {
        subgroup.Delete();
    }
}
```

5.11.4.9 Technologieobjekt übersetzen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)
- Das Technologieobjekt ist vorhanden.
Siehe Technologieobjekt erstellen (Seite 550)

Programmcode: Übersetzen eines Technologieobjekts

Um ein Technologieobjekt zu übersetzen, ändern Sie den folgenden Programmcode:

```
// Compile a single technology object
private static void CompileSingleTechnologicalObject(TechnologicalInstanceDB
technologicalObject)
{
    ICompilable singleCompile = technologicalObject.GetService<ICompilable>();
    CompilerResult compileResult = singleCompile.Compile();
}
```

Programmcode: Übersetzen der Technologieobjektgruppe

Um die Technologieobjektgruppe zu übersetzen, ändern Sie den folgenden Programmcode:

```
// Compile technology object group
private static void CompileTechnologicalObjectGroup(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBGGroup technologicalObjectGroup =
plcSoftware.TechnologicalObjectGroup;
    ICompilable groupCompile = technologicalObjectGroup.GetService<ICompilable>();
    CompilerResult compileResult = groupCompile.Compile();
}
```

Übersetzungsergebnisse

Die Übersetzungsergebnisse von Technologieobjekten werden rekursiv gespeichert.

Ein Beispiel für die rekursive Auswertung von Übersetzungsergebnissen finden Sie in Abschnitt "Projekt übersetzen (Seite 157)".

Hinweis**Weitere Parameter**

Importierte Daten werden nach dem Übersetzen erweitert. Sie können Openness beispielsweise verwenden, um Technologieobjekte für die Bewegungssteuerung mit unterschiedlichen Versionen zu erstellen. Nach dem Übersetzen von Technologieobjekten werden die Bewegungssteuerungsversionen anderer Technologieobjekte an diejenigen des zuletzt erstellten Technologieobjekts angepasst.

Siehe auch

Standard-Bibliotheken (Seite 76)

5.11.4.10 Technologieobjekte enumerieren

Voraussetzung

Sie können die TOs der obersten Ebene wie "TO_Axis" und "TO_Cam" enumerieren. Um untergeordnete TOs wie "TO_OutputCam" zu finden, siehe TO_OutputCam, TO_CamTrack und TO_MeasuringInput erstellen und finden (Seite 569).

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)

Programmcode

Um Technologieobjekte zu enumerieren, ändern Sie den folgenden Programmcode:

```
// Enumerate all technology objects
private static void EnumerateTechnologicalObjects(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBComposition technologicalObjects =
    plcSoftware.TechnologicalObjectGroup.TechnologicalObjects;
    foreach (TechnologicalInstanceDB technologicalObject in technologicalObjects)
    {
        // Do something ...
    }
}
```

Siehe auch

Standard-Bibliotheken (Seite 76)
S7-1500 Motion Control (Seite 569)

5.11.4.11 Gruppen für Technologieobjekte enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)

Verwendung

Enthaltene Untergruppen werden beim Enumerieren rekursiv berücksichtigt.

Programmcode: Alle Gruppen enumerieren

Um die Gruppen für Technologieobjekte zu enumerieren, ändern Sie folgenden Programmcode:

```
//Enumerates all groups for technology objects
private static void EnumerateRecursively(PlcSoftware sw)
{
    foreach (TechnologicalInstanceDBUserGroup group in sw.TechnologicalObjectGroup.Groups)
    {
        EnumerateRecursivelySubGroups (group);
    }
}

//Enumerates all included sub groups
private static void EnumerateRecursivelySubGroups(TechnologicalInstanceDBUserGroup group)
{
    foreach (TechnologicalInstanceDBUserGroup subGroup in group.Groups)
    {
        EnumerateRecursivelySubGroups (subGroup);
        // recursion
    }
}
```

Programmcode: Auf eine Gruppe zugreifen

Um auf eine ausgewählte benutzerdefinierte Gruppe für Technologieobjekte zuzugreifen, ändern Sie folgenden Programmcode:

```
//Gives individual access to a specific group for technology objects
private static void AccessTechnologicalInstanceDBUserGroup(PlcSoftware sw)
{
    TechnologicalInstanceDBUserGroupComposition groups =
sw.TechnologicalObjectGroup.Groups;
    TechnologicalInstanceDBUserGroup userGroup = groups.Find("MyUserfolder");
}
```


5.11.4.12 Technologieobjekt finden

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)

Programmcode

Um ein spezifisches Technologieobjekt zu finden, ändern Sie den folgenden Programmcode:

```
// Find a specific technology object by its name
private static void FindTechnologicalObject(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBComposition technologicalObjects =
    plcSoftware.TechnologicalObjectGroup.TechnologicalObjects;
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject = technologicalObjects.Find(nameOfTO);
}
```

Siehe auch

Standard-Bibliotheken (Seite 76)

5.11.4.13 Parameter eines Technologieobjekts enumerieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)
- Ein Technologieobjekt ist vorhanden.
Siehe Technologieobjekt erstellen (Seite 550) oder Parameter eines Technologieobjekts finden (Seite 558)
- Das Technologieobjekt (Seite 547) unterstützt diese Funktion.

Programmcode

Um Parameter eines bestimmten Technologieobjekts zu enumerieren, ändern Sie den folgenden Programmcode:

```
// Enumerate parameters of a technology object
private static void EnumerateParameters(PlcSoftware plcSoftware)
{
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find(nameOfTO);

    foreach (TechnologicalParameter parameter in technologicalObject.Parameters)
    {
        // Do something ...
    }
}
```

Siehe auch

Standard-Bibliotheken (Seite 76)

Technologieobjekt finden (Seite 557)

5.11.4.14 Parameter eines Technologieobjekts finden

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe [Verbindung zum TIA Portal aufbauen](#) (Seite 84)
- Ein Projekt ist geöffnet.
Siehe [Projekt öffnen](#) (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe [PLC-Target und HMI-Target abfragen](#) (Seite 277)
- Ein Technologieobjekt liegt vor.
Siehe [Technologieobjekt erstellen](#) (Seite 550)
- Das Technologieobjekt (Seite 547) unterstützt diese Funktion.

Programmcode

Um Parameter eines bestimmten Technologieobjekts zu finden, ändern Sie den folgenden Programmcode:

```
// Find parameters of a technology object
private static void FindParameterOfTechnologicalObject(PlcSoftware plcSoftware)
{
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find(nameOfTO);

    string nameOfParameter = "Config.InputUpperLimit";
    TechnologicalParameter parameter =
technologicalObject.Parameters.Find(nameOfParameter);
}
```

Parameter verschiedener Technologieobjekte

Parameter von SIMATIC S7-1200 Motion Control (Seite 561)

Parameter von SIMATIC S7-1500 Motion Control (Seite 569)

Parameter der PID-Regelung (Seite 599)

Zählparameter (Seite 599)

Parameter von Easy Motion Control (Seite 600)

Siehe auch

Standard-Bibliotheken (Seite 76)

Technologieobjekt finden (Seite 557)

5.11.4.15 Parameter eines Technologieobjekts lesen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)
- Ein Technologieobjekt liegt vor.
Siehe Technologieobjekt erstellen (Seite 550)
- Das Technologieobjekt (Seite 547) unterstützt diese Funktion.

Programmcode

Um Parameter eines bestimmten Technologieobjekts zu lesen, ändern Sie den folgenden Programmcode:

```
// Read parameters of a technology object
private static void ReadParameterOfTechnologicalObject(PlcSoftware plcSoftware)
{
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find(nameOfTO);

    string nameOfParameter = "Config.InputUpperLimit";
    TechnologicalParameter parameter =
technologicalObject.Parameters.Find(nameOfParameter);

    // Read from parameter
    string name = parameter.Name;
    object value = parameter.Value;
}
```

Siehe auch

Standard-Bibliotheken (Seite 76)

Technologieobjekt finden (Seite 557)

5.11.4.16 Parameter eines Technologieobjekts schreiben

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PLC ist im Projekt ermittelt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)
- Ein Technologieobjekt liegt vor.
Siehe Technologieobjekt erstellen (Seite 550)
- Das Technologieobjekt (Seite 547) unterstützt diese Funktion.

Ausnahme

Eine `EngineeringException` wird gemeldet, wenn:

- Sie einen neuen Wert für einen Parameter festlegen, der keinen Schreibzugriff bietet.
- Ein neuer Wert für einen Parameter einen nicht unterstützten Typ hat.

Programmcode

Um Parameter eines spezifischen Technologieobjekts zu schreiben, ändern Sie den folgenden Programmcode:

```
// Write parameters of a technology object
private static void WriteParameterOfTechnologicalObject(PlcSoftware plcSoftware)
{
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find(nameOfTO);

    string nameOfParameter = "Config.InputUpperLimit";
    TechnologicalParameter parameter =
technologicalObject.Parameters.Find(nameOfParameter);

    // Write to parameter if the value is writable
    object value = 3.0;
    parameter.Value = value;
}
```

Parameter der verschiedenen Technologieobjekte

Parameter von SIMATIC S7-1200 Motion Control (Seite 561)

Parameter von SIMATIC S7-1500 Motion Control (Seite 569)

Parameter der PID-Regelung (Seite 599)

Zählparameter (Seite 599)

Parameter von Easy Motion Control (Seite 600)

Siehe auch

Standard-Bibliotheken (Seite 76)

Technologieobjekt finden (Seite 557)

5.11.4.17 S7-1200 Motion Control

PTO-Ausgänge verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131).

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

- Eine S7-1200 PLC mit PTO-Ausgängen ist im Projekt angelegt.
- Das Technologieobjekt ist vorhanden.
Siehe Technologieobjekt erstellen (Seite 550).

Programmcode

Um einen PTO-Ausgang für das "TO_PositioningAxis" zu verschalten, ändern Sie folgenden Programmcode:

```
//An instance of the technology object axis is already available in the program before
private static void ConnectingDrive(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to PTO mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 2;
    //Set PTO-Output
    technologicalObject.Parameters.Find("_Actor.Interface.PTO").Value = 0; // select
Pulse_1
    // 0 = Pulse_1
    // 1 = Pulse_2
    // 2 = Pulse_3
    // 3 = Pulse_4
}
```

PROFIdrives mit der Hardware-Adresse verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein PROFIdrive ist in dem Projekt verfügbar und mit der S7-1200 CPU verschaltet.
- Das Technologieobjekt liegt vor.
Siehe Technologieobjekt erstellen (Seite 550).

Programmcode

Ändern Sie den folgenden Programmcode, um einen PROFIdrive durch eine Hardware-Adresse mit „TO_PositioningAxis“ zu verschalten.

```
//An instance of the technology object axis is already available in the program before
private static void ConnectingDrive(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to PROFIdrive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 1;

    //Set axis to drive mode
    technologicalObject.Parameters.Find("_Actor.Interface.DataConnection").Value = 0;

    //Set connection to address of drive. The output will be set automatically.
    technologicalObject.Parameters.Find("_Actor.Interface.ProfiDriveIn").Value = "%I68.0";
    technologicalObject.Parameters.Find("Sensor[1].Interface.Number").Value = 1;
    // 1 = Encoder1, 2 = Encoder2;
}
```

Geber für PROFIdrives mit der Hardware-Adresse verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein PROFIdrive ist in dem Projekt verfügbar und mit der S7-1200 CPU verschaltet.
- Das Technologieobjekt liegt vor.
Siehe Technologieobjekt erstellen (Seite 550).

Programmcode

Ändern Sie den folgenden Programmcode, um einen Geber mit der Hardware-Adresse mit dem "TO_PositioningAxis" zu verschalten:

```
//An instance of the technology object axis is already available in the program before
private static void ConnectingEncoder(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to PROFIdrive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 1;

    //Set the encoder mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.EncoderConnection").Value =
7;

    //Set axis to use PROFINET encoder
    technologicalObject.Parameters.Find("_Sensor[1].Interface.DataConnection").Value = 0;

    //Set connection to address of drive. The output will be set automatically.
    technologicalObject.Parameters.Find("_Sensor[1].Interface.ProfiDriveIn").Value =
"%I68.0";
    technologicalObject.Parameters.Find("Sensor[1].Interface.Number").Value = 1;
    // 1 = Encoder1, 2 = Encoder2;
}
```

Analogantriebe mit der Hardware-Adresse verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein Analogantrieb ist in dem Projekt verfügbar und mit der S7-1200 CPU verschaltet.
- Das Technologieobjekt liegt vor.
Siehe Technologieobjekt erstellen (Seite 550).

Programmcode

Ändern Sie den folgenden Programmcode, um einen Analogantrieb mit der Hardware-Adresse mit dem "TO_PositioningAxis" zu verschalten:

```
//An instance of the technology object axis is already available in the program before
private static void ConnectingEncoder(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to analog drive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 0;

    //Set axis to drive mode
    technologicalObject.Parameters.Find("_Actor.Interface.DataConnection").Value = 0;

    //Set connection to analog address of drive
    technologicalObject.Parameters.Find("_Actor.Interface.Analog").Value = "%QW64";
}
```

Geber für Analogantriebe mit der Hardware-Adresse verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein Analogantrieb ist in dem Projekt verfügbar und mit der S7-1200 CPU verschaltet.
- Das Technologieobjekt liegt vor.
Siehe Technologieobjekt erstellen (Seite 550).

Programmcode

Ändern Sie den folgenden Programmcode, um einen Geber mit der Hardware-Adresse mit dem "TO_PositioningAxis" zu verschalten:

```
//An instance of the technology object axis is already available in the program before
//Connecting by High Speed Counter mode
private static void ConnectingEncoder(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to analog drive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 0;

    //Set encoder for high-speed counter mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.EncoderConnection").Value =
4;
    technologicalObject.Parameters.Find("_Sensor[1].Interface.HSC.Name").Value = "HSC_1";
}

//An instance of the technology object axis is already available in the program before
//Connecting by PROFINET/PROFIBUS telegram
private static void ConnectingEncoder(TechnologicalInstanceDB
technologicalObject)
{
    //Set axis to analog drive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 0;
    //Set encoder for PROFINET/PROFIBUS mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.EncoderConnection").Value =
7;
    technologicalObject.Parameters.Find("_Sensor[1].Interface.DataConnection").Value =
"Encoder";
    technologicalObject.Parameters.Find("_Sensor[1].Interface.ProfiDriveIn").Value =
"%I68.0";
    technologicalObject.Parameters.Find("Sensor[1].Interface.Number").Value = 1;
    // 1 = Encoder1, 2 = Encoder2;
}
```

Antriebe mit einem Datenbaustein verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1200 CPU ist im Projekt angelegt.

- Ein Datenbaustein ist in dem Projekt verfügbar und auf "Nicht optimiert" eingestellt
Bei einem PROFIdrive-Achstyp enthält der Datenbaustein eine Variable dieses Typs, z. B. PD_TEL3.
Bei einem Analogantrieb enthält der Datenbaustein eine Variable des Datentyps Wort.
- Das Technologieobjekt liegt vor.
Siehe Technologieobjekt erstellen (Seite 550).

Programmcode

Ändern Sie den folgenden Programmcode, um einen PROFIdrive mit einem Datenbaustein mit dem "TO_PositioningAxis" zu verschalten.

```
//An instance of the technology object axis is already available in the program before
private static void ConfigureDrivewithDataBlock(TechnologicalInstanceDB
technologicalObject)
{
    //Set axis to PROFIdrive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 1;

    //Set axis to data block mode
    technologicalObject.Parameters.Find("_Actor.Interface.DataConnection").Value = 1;

    //Set the tag in the data block
    technologicalObject.Parameters.Find("_Actor.Interface.DataBlock").Value =
    "Data_block_1.Member_of_type_PD_TEL3";
}
```

Programmcode

Ändern Sie den folgenden Programmcode, um einen Analogantrieb mit einem Datenbaustein mit dem "TO_PositioningAxis" zu verschalten:

```
//An instance of the technology object axis is already available in the program before
//Connecting an analog drive with data block.
private static void ConfigureDrivewithDataBlock(TechnologicalInstanceDB
technologicalObject)
{
    //Set axis to analog mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 0;

    //Set the tag in the data block
    technologicalObject.Parameters.Find("_Actor.Interface.Analog").Value =
    "Data_block_1.Static_1";
}
```

Geber mit einem Datenbaustein verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein Datenbaustein ist in dem Projekt verfügbar und auf "Nicht optimiert" eingestellt
Bei PROFIdrive enthält der Datenbaustein eine Variable dieses Typs, z. B. PD_TEL3.
- Das Technologieobjekt liegt vor.
Siehe Technologieobjekt erstellen (Seite 550).

Programmcode

Ändern Sie den folgenden Programmcode, um einen Geber mit einem Datenbaustein zu verschalten:

```
//An instance of the technology object axis is already available in the program before
private static void ConfigureEncoderwithDataBlock(TechnologicalInstanceDB
technologicalObject)
{
    //Set axis to PROFIdrive mode depending by axis type. 1 = PROFIdrive, 0 = Analog Drive.
    technologicalObject.Parameters.Find("Actor.Type").Value = 1;

    //Set the encoder mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.EncoderConnection").Value =
7;

    //Set axis to data block mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.DataConnection").Value = 1;

    //Set the tag in the data block. For PD_TEL3 and PD_TEL4 "Encoder1" or "Encoder2".
    technologicalObject.Parameters.Find("_Sensor[1].Interface.DataBlock").Value =
>Data_block_1.Member_of_Type_PD_TEL3";
}
```

Parameter für TO_PositioningAxis und TO_CommandTable

Eine Liste mit allen verfügbaren Variablen finden Sie im Funktionshandbuch SIMATIC STEP 7 S7-1200 Motion Control im Internet (<https://support.industry.siemens.com/cs/ww/de/view/109773400>).

Hinweis

Im TIA Portal finden Sie in der Parametersicht der Technologieobjektconfiguration die Spalte "Name in Openness".

5.11.4.18 S7-1500 Motion Control

TO_OutputCam, TO_CamTrack und TO_MeasuringInput erstellen und finden

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_PositioningAxis, TO_SynchronousAxis oder TO_ExternalEncoder ist im Projekt angelegt.

Anwendung

Die Technologieobjekte Nocken, Nockenspur und Messtaster sind mit einem der Technologieobjekte Positionierachse, Gleichlaufachse oder Externer Geber verschaltet. Um auf die Technologieobjekte Nocken, Nockenspur oder Messtaster zuzugreifen, verwenden Sie den Service OutputCamMeasuringInputContainer.

Programmcode: Technologieobjekte Nocken, Nockenspur und Messtaster erstellen und finden

Ändern Sie den folgenden Programmcode, um das Technologieobjekt Nocken, Nockenspur oder Messtaster zu erstellen oder zu finden.

```
/*An instance of the technology object under which the TO_OutputCam, TO_CamTrack or
TO_MeasuringInput should be created is already available in the program before*/
private static void CreateFind_OutputcamCamtrackMeasuringinput (TechnologicalInstanceDB
technologyObject)
{
    //Retrieve service OutputCamMeasuringInputContainer
    OutputCamMeasuringInputContainer container =
    technologyObject.GetService<OutputCamMeasuringInputContainer>();
    //Get access to TO_OutputCam / TO_CamTrack container
    TechnologicalInstanceDBComposition outputcamCamtrackContainer = container.OutputCams;

    //Find technology object TO_OutputCam or TO_CamTrack
    TechnologicalInstanceDB outputCam = outputcamCamtrackContainer.Find("OutputCamName");
    TechnologicalInstanceDB camTrack = outputcamCamtrackContainer.Find("CamTrackName");

    //Create new technology object TO_OutputCam or TO_CamTrack
    TechnologicalInstanceDB newOutputCam =
    outputcamCamtrackContainer.Create("NewOutputCamName", "TO_OutputCam",
    new Version(3, 0));
    TechnologicalInstanceDB newCamTrack =
    outputcamCamtrackContainer.Create("NewCamTrackName", "TO_CamTrack", new Version(3, 0));

    //Get access to TO_MeasuringInput container
    TechnologicalInstanceDBComposition measuringInputContainer = container.MeasuringInputs;

    //Find technology object TO_MeasuringInput
    TechnologicalInstanceDB measuringInput =
    measuringInputContainer.Find("MeasuringInputName");

    //Create new technology object TO_MeasuringInput
    TechnologicalInstanceDB newMeasuringInput =
    measuringInputContainer.Create("NewMeasuringInput", "TO_MeasuringInput",
    new Version(3, 0));
}
```

Sie erstellen die S7-1500 Technologieobjekte OutputCam, CamTrack und MeasuringInput als Masterkopie von Siemens.Engineering.Library.MasterCopies.MasterCopy.

```
/*An instance of the technology object under which the TO_OutputCam, TO_CamTrack or
TO_MeasuringInput should be created is already available in the program before*/
//_sourceMasterCopy contains master copy object output cam
private static void CreateFind_OutputcamCamtrackMeasuringinput (TechnologicalInstanceDB
technologyObject, MasterCopy sourceMasterCopy)
{
    //Retrieve service OutputCamMeasuringInputContainer
    OutputCamMeasuringInputContainer container =
    technologyObject.GetService<OutputCamMeasuringInputContainer>();

    TechnologicalInstanceDB result = container.OutputCams.CreateFrom(sourceMasterCopy);
}
```

Parameter von S7-1500 Motion Control

Parameter von S7-1500 Motion Control

Die meisten Parameter von S7-1500 Motion Control Technologieobjekten sind direkt auf Datenbausteinvariablen abgebildet, doch es gibt auch einige zusätzliche Parameter, die nicht direkt auf Datenbausteine abgebildet sind.

Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Sie haben Zugriff auf alle Technologieobjekt-Datenbausteinvariablen, die im Allgemeinen beschrieben sind, außer:

- Schreibgeschützte Variablen
- Variablen des Datentyps VREF
- Variablen mit der Struktur „InternalToTrace“
- Variablen mit der Struktur „ControlPanel“

Einige Technologieparameter, die auf schreibgeschützte Datenbausteinvariablen abgebildet sind, sind in der PublicAPI schreibbar. Die zulässigen Werte sind die gleichen wie die Werte für die zugrundeliegenden Datenbausteinvariablen.

In den folgenden Kapiteln sind die betroffenen Parameter für alle Technologieobjekte aufgelistet.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Programmcode

Ändern Sie den folgenden Programmcode, um auf die direkt abgebildeten Parameter zuzugreifen:

```
//An instance of the technology object is already available in the program before
private static void ReadWriteDataBlockTag(TechnologicalInstanceDB technologyObject)
{
    //Read value from data block tag "ReferenceSpeed"
    double value =
        (double)technologyObject.Parameters.Find("Actor.DriveParameter.ReferenceSpeed").Value;

    //Write data block tag "ReferenceSpeed"
    technologyObject.Parameters.Find("Actor.DriveParameter.ReferenceSpeed").Value = 3000.0;
}
```

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

S7-1500 Motion Control-Technologieobjekte sind mit anderen Geräten oder Technologieobjekten verschaltet. Die Informationen zu diesen Verschaltungen werden in zusätzlichen Parametern sowie in Informationen zu Maßeinheiten gespeichert. Zusätzliche Parameter sind nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet.

In den folgenden Kapiteln sind die betroffenen Parameter für alle Technologieobjekte aufgelistet.

Programmcode: Zusätzliche Parameter

Ändern Sie den folgenden Programmcode, um auf die zusätzlichen Parameter zuzugreifen:

```
//An instance of the technology object is already available in the program before
private static void ReadWriteAdditionalParameter(TechnologicalInstanceDB technologyObject)
{
    //Read additional parameter "_Properties.MotionType"
    uint value = (uint)technologyObject.Parameters.Find("_Properties.MotionType").Value;

    //Write additional parameter "_Properties.MotionType"
    technologyObject.Parameters.Find("_Properties.MotionType").Value = 1;
}
```

Weitere Informationen

Weitere Informationen finden Sie in den Funktionshandbüchern SIMATIC S7-1500 Motion Control:

<https://support.industry.siemens.com/cs/ww/de/view/109751049> (<https://support.industry.siemens.com/cs/ww/de/view/109751049>)

Achse und Geber

Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Bei den Technologieobjekten Achse und Externer Geber sind die folgenden schreibgeschützten Datenbausteinvariablen in der PublicAPI schreibbar.

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness	TO_SpeedAxis	TO_PositioningAxis TO_SynchronousAxis	TO_ExternalEncoder
Actor.Type	Antriebstyp	0: Analogausgang 1: PROFIdrive-Telegramm	int	X	X	-
Actor.Interface.EnableDriveOutput	"Freigabeausgang" für Analogantriebe	True, False	bool	X	X	-
Actor.Interface.DriveReadyInput	"Bereiteingang" für Analogantriebe	True, False	bool	X	X	-
Actor.Interface.EnableTorqueData	Drehmomentdaten	True, False	bool	-	X	-
VirtualAxis.Mode	Virtuelle Achse	0, 1	uint	X	X	-
Sensor[n].Existent ¹⁾	Geber verwenden	True, False	bool	-	X	-
Sensor[n].Interface.Number ¹⁾	Nummer des Gebers im Telegramm	1 ... 2	uint	-	X	-
Sensor[n].Type ¹⁾	Gebertyp	0: Inkrementell 1: Absolut 2: Zyklisch absolut	int	-	X	-
Sensor.Interface.Number	Nummer des Gebers im Telegramm	1 ... 2	uint	-	-	X
Sensor.Type	Gebertyp	0: Inkrementell 1: Absolut 2: Zyklisch absolut	int	-	-	X
CrossPlcSynchronous-Operation.Interface[n].EnableLeadingValueOutput ⁵⁾	PLC-übergreifenden Leitwert bereitstellen	Wahr, Falsch	bool	-	X	X

¹⁾ S7-1500 CPU: n=1; S7-1500T CPU: 1≤n≤4

⁵⁾ V5.0 n=1; ≥V6.0 1≤n≤8

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Bei den Technologieobjekten Achse und Externer Geber sind die folgenden zusätzlichen Parameter verfügbar.

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness	TO_SpeedAxis	TO_PositioningAxis TO_SynchronousAxis	TO_ExternalEncoder
_Actor.DataAdaptionOffline	Automatisch anwenden...	Wahr, Falsch	bool	X	X	-
_Actor.Interface.Telegram	Antriebstelegramm	Telegramm Nummer ²⁾	uint	X	X	-
_Actor.Interface.EnableDriveOutputAddress	Adresse für Ausgang Antriebsfreigabe	PublicAPI-Objekt	SW.Tags.PlcTag	X	X	-
_Actor.Interface.DriveReadyInputAddress	Adresse für Eingang Antrieb bereit	PublicAPI-Objekt	SW.Tags.PlcTag	X	X	-
_Units.LengthUnit	Positionseinheiten	Siehe Variable Units.LengthUnit ³⁾	uint	-	X	X
_Units.VelocityUnit	Geschwindigkeitseinheiten	Siehe Variable Units.VelocityUnit ³⁾	uint	X	X	X
_Units.TorqueUnit	Drehmomentseinheiten	Siehe Variable Units.TorqueUnit ³⁾	uint	X	X	-
_Units.ForceUnit	Krafteinheiten	Siehe Variable Units.ForceUnit ³⁾	uint	-	X	-
_Sensor[n].DataAdaptionOffline ¹⁾	Automatisch anwenden...	Wahr, Falsch	bool	-	X	-
_Sensor[n].Interface.Telegram ¹⁾	Gebertelegramm	Telegramm Nummer ²⁾	uint	-	X	-
_Sensor[n].ActiveHoming.DigitalInputAddress ¹⁾	Digitaleingang	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	-
_Sensor[n].PassiveHoming.DigitalInputAddress ¹⁾	Digitaleingang	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	-
_Sensor.DataAdaptionOffline	Automatisch anwenden...	Wahr, Falsch	bool	-	-	X
_Sensor.Interface.Telegram	Gebertelegramm	Telegramm Nummer ²⁾	uint	-	-	X
_Sensor.ActiveHoming.DigitalInputAddress	Digitaleingang	PublicAPI-Objekt	SW.Tags.PlcTag	-	-	X
_Sensor.PassiveHoming.DigitalInputAddress	Digitaleingang	PublicAPI-Objekt	SW.Tags.PlcTag	-	-	X

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness	TO_SpeedAxis	TO_PositioningAxis TO_SynchronousAxis	TO_ExternalEncoder
_Properties.Motion-Type	Achstyp bzw. "Technische Einheit der Position"	0: Linear 1: Rotatorisch	int	-	X	X
_Properties.UseHigh-ResolutionPositionValues ⁴⁾	Positionswerte mit höherer Auflösung verwenden	True, False	bool	-	X	X
_PositionLimits_HW.MinSwitchAddress	Adresse negativer Hardware-Endschalter	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	-
_PositionLimits_HW.MaxSwitchAddress	Adresse positiver Hardware-Endschalter	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	-
_CrossPlcSynchronousOperation.Interface[n]. AddressOut ⁵⁾	Ausgangsadresse für das Leitwerttelegramm	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	X
_CrossPlcSynchronousOperation.ActivateLocalLeading-ValueDelayTimeCalculation ⁴⁾	Systemseitige Berechnung zulassen	True, False	bool	-	X	X

¹⁾ S7-1500 CPU: n=1; S7-1500T CPU: 1≤n≤4

²⁾ Mögliche Werte sind im Funktionshandbuch S7-1500 Motion Control im Kapitel PROFIdrive-Telegramme beschrieben

³⁾ Mögliche Werte sind im Funktionshandbuch S7-1500 Motion Control im Kapitel "Variablen Units (TO)" beschrieben

⁴⁾ ≥ V5.0

⁵⁾ V5.0 n=1; ≥V6.0 1≤n≤8

Leitachsstellvertreter

Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Beim Technologieobjekt Leitachsstellvertreter sind keine schreibgeschützten Datenbausteinvariablen in der PublicAPI schreibbar.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Beim Technologieobjekt Leitachsstellvertreter sind die folgenden zusätzlichen Parameter verfügbar.

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp
_Interface.AddressIn ¹⁾	Übertragungsbereich	PublicAPI-Objekt	SW.Tags.PlcTag

¹⁾ S7-1500T CPU

Messtaster**Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind**

Beim Technologieobjekt Messtaster sind die folgenden schreibgeschützten Datenbausteinvariablen in der PublicAPI schreibbar.

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness
Parameter.MeasuringInput-Type	Messtastertyp	0: Messen über Zeitstempel (STANDARD) 1: Messen über PROFIdrive-Telegramme des Antriebs oder externen Gebers (PROFIDRIVE) 2: Messen durch Abhören des Messtasters ¹⁾	Int

¹⁾ ≥ V8.0

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Beim Technologieobjekt Messtaster sind die folgenden zusätzlichen Parameter verfügbar:

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness
_AssociatedObject	Zugeordnete Achse oder Externer Geber	PublicAPI-Objekt	SW.TechnologicalObjects.TechnologicalInstanceDB
_ListenToMeasuringInput	Abhören des Messtasters	Technologieobjekte vom Typ TO_MeasuringInput	SW.TechnologicalObjects.TechnologicalInstanceDB

Nocken und Nockenspur

Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Beim Technologieobjekt Nocken sind die folgenden schreibgeschützten Datenbausteinvariablen in der PublicAPI schreibbar.

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness
Interface.LogicOperation	Logische Verknüpfung des Nockensignals am Ausgang	0: ODER-Verknüpfung 1: UND-Verknüpfung	int

Beim Technologieobjekt Nockenspur sind keine schreibgeschützten Datenbausteinvariablen in der PublicAPI schreibbar.

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Für die Technologieobjekte Nocken und Nockenspur sind die folgenden zusätzlichen Parameter verfügbar:

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness
_AssociatedObject	Zugeordnete Achse oder Ex- terner Geber	PublicAPI-Objekt	SW.TechnologicalObjects.Techno- logicalInstanceDB

Kinematik

Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Beim Technologieobjekt Kinematik sind die folgenden schreibgeschützten Datenbausteinvariablen in der PublicAPI schreibbar.

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness
Kinematics.TypeOfKinematics	Kinematiktyp	Siehe Variable Kinema- tics.TypeOfKinematics ¹⁾	int
MotionQueue.MaxNumber- OfCommands	Maximale Anzahl der Aufträ- ge	1..10	Int

¹⁾ Mögliche Werte sind im versionsspezifischen Funktionshandbuch S7-1500T Kinematikfunktionen im Kapitel "Variable Kinematics" beschrieben.

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Beim Technologieobjekt Kinematik sind die folgenden zusätzlichen Parameter verfügbar:

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp
_KinematicsAxis[n] ¹⁾	Kinematikachse A1 ... A6	Achse, die mit TO_Kine- matics-Objekten ver- schaltet werden kann	SW.TechnologicalObjects.Techno- logicalInstanceDB
_Units.LengthUnit	Maßeinheit > Position	Siehe Variable Units.LengthUnit ²⁾	uint

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp
_Units.LengthVelocityUnit	Maßeinheit > Geschwindigkeit	Siehe Variable Units.LengthVelocityUnit ²⁾	uint
_Units.AngleUnit	Maßeinheit > Winkel	Siehe Variable Units.AngleUnit ²⁾	uint
_Units.AngleVelocityUnit	Maßeinheit > Winkelgeschwindigkeit	Siehe Variable Units.AngleVelocityUnit ²⁾	uint
_Properties.UseHighResolutionPositionValues ¹⁾	Positionswerte mit höherer Auflösung verwenden	True, False	bool
_X_Minimum ³⁾	x-Minimum	-1.00E+12 .. +1.00E+12	double
_X_Maximum ³⁾	x-Maximum	-1.00E+12 .. +1.00E+12	double
_Y_Minimum ³⁾	y-Minimum	-1.00E+12 .. +1.00E+12	double
_Y_Maximum ³⁾	y-Maximum	-1.00E+12 .. +1.00E+12	double
_Z_Minimum ³⁾	z-Minimum	-1.00E+12 .. +1.00E+12	double
_Z_Maximum ³⁾	z-Maximum	-1.00E+12 .. +1.00E+12	double
_A3_Maximum ³⁾	A3-Maximum	-1.00E+12 .. +1.00E+12	double

¹⁾ < V7.0 $n = 1 \leq n \leq 4$; $\geq V7.0$ $1 \leq n \leq 6$; Kinematikachsen A5 und A6 nur mit "S7-1500T Motion Control KinPlus" verwendbar

²⁾ Mögliche Werte sind im versionsspezifischen Funktionshandbuch S7-1500T Kinematikfunktionen im Kapitel "Variable Units" beschrieben.

³⁾ $\geq V5.0$

Interpreter

Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Beim Technologieobjekt Interpreter sind die folgenden schreibgeschützten Datenbausteinvariablen in der PublicAPI schreibbar.

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness
Parameter.MaxNumberOfCommands	Maximale Anzahl der Aufträge	10..100	Int

Antriebe verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500 CPU ist im Projekt angelegt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277).

- Ein Technologieobjekt des Typs `TO_SpeedAxis`, `TO_PositioningAxis` oder `TO_SynchronousAxis` ist im Projekt angelegt.
- Ein Antrieb ist im Projekt angelegt.

Anwendung

Um eine Achse mit einem Antrieb zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben.

Sensorschnittstelle und Telegramm 750

- Wenn die Aktorschnittstelle eine Sensorschnittstelle enthält, wird die Sensorschnittstelle automatisch über diese Methoden verschaltet.
- Enthält die Aktorschnittstelle Drehmomentdaten, wird das Zusatztelegramm 750 automatisch über diese Methoden verschaltet.
- Um auf die Sensorschnittstelle zuzugreifen, können Sie `SensorInterface[m]` mit $0 \leq m \leq 3$ verwenden.

Der Typ der Public API `AxisEncoderHardwareConnectionInterface` bietet die folgenden Methoden, die zur Verschaltung oder Trennung der Aktor- oder Sensorschnittstellen verwendet werden können:

Methode	Beschreibung
<code>void Connect(HW.DevicItem moduleInOut)</code>	Verschaltet mit Eingangs- und Ausgangsadressen am gleichen Modul. Verwenden Sie diese Methode zur Verschaltung von Antrieben, die mit Hilfe einer GSD-Datei projiziert werden.
<code>void Connect(HW.DevicItem moduleIn, HW.DevicItem moduleOut)</code>	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen Verwenden Sie diese Methode zur Verschaltung von Antrieben, die mit Hilfe einer GSD-Datei projiziert werden.
<code>void Connect(HW.DevicItem moduleIn, HW.DevicItem moduleOut, ConnectOption connectOption)</code>	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen unter Angabe einer zusätzlichen <code>ConnectOption</code> . Verwenden Sie diese Methode zur Verschaltung von Antrieben, die mit Hilfe einer GSD-Datei projiziert werden.
<code>void Connect(HW.Channel channel)</code>	Verschaltet mit einem Kanal Verwenden Sie diese Methode zur Verschaltung von Technologiemodulen.
<code>void Connect(MC.Drives.Telegram telegram)</code>	Verschaltet mit einem vom Antriebsobjekt <code>StartDrive</code> bereitgestellten Telegramm Verwenden Sie diese Methode zur Verschaltung von Antrieben, die mit Startdrive ab V19 projiziert werden.
<code>void Connect(MC.Drives.Telegram telegram, ConnectOption connectOption)</code>	Verschaltet mit einem vom Antriebsobjekt <code>StartDrive</code> bereitgestellten Telegramm und gibt eine zusätzliche <code>ConnectOption</code> an. Verwenden Sie diese Methode zur Verschaltung von Antrieben, die mit Startdrive ab V19 projiziert werden.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Methode	Beschreibung
<code>void Connect(int addressIn, int addressOut, ConnectOption connectOption)</code>	Verschaltet Bit-Adressen direkt Diese Methode können Sie in allen Fällen verwenden. Wird der Antrieb mit Startdrive < V19 projektiert, muss diese Methode verwendet werden. Lesen Sie die Byteadresse des Telegramms aus der Hardware-konfiguration aus und verwenden Sie die entsprechende Bit-adresse. $\text{Bitadresse} = \text{Byteadresse} * 8$
<code>void Connect(string pathToDBMember)</code>	Verschaltet mit einer Datenbausteinvariablen Verwenden Sie diese Methode zur Verschaltung einer Achse mit einem Antrieb über einen Datenbaustein. Der Datenbaustein für die Verschaltung muss vorhanden sein.
<code>void Connect(SW.Tags.PlcTag outputTag)</code>	Verschaltet mit einer CPU-Variablen Verwenden Sie diese Methode zur Verschaltung eines Antriebs mit analoger Sollwertschnittstelle mit Variablen. Sie können Variablen für "Analogausgang", "Freigabeausgang" und "Bereiteingang" verschalten.
<code>void Disconnect()</code>	Trennt eine existierende Verschaltung

ET200SP PTO 2

Bei der Verschaltung einer Baugruppe ET200SP PTO 2 mit einem Antrieb durch Direktvorgabe der Bitadressen müssen folgende Offsets verwendet werden:

Kanal	Telegramm	Bitoffset zu Eingangsadresse	Bitoffset zu Ausgangsadresse
0	1	0	0
	81	32	32
1	1	128	64
	81	160	96

Attribut

Um festzustellen, wie das Technologieobjekt verschaltet wird, können Sie die folgenden schreibgeschützten Attribute verwenden. Die entsprechenden Verschaltungswerte werden nur eingestellt, wenn eine Verschaltung dieser spezifischen Art besteht.

Attribut	Datentyp	Beschreibung
IsConnected	bool	TRUE: Schnittstelle ist verschaltet FALSE: Schnittstelle ist nicht verschaltet
InputOutputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangs- und Ausgangsadressen enthält
InputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangsadressen enthält Der Wert wird auch im Falle einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
OutputModule	HW.DeviceItem	Verschaltetes Modul, das Ausgangsadressen enthält Der Wert wird auch im Falle einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
InputAddress	int	Logische Eingangsadresse des verschalteten Objekts, beispielsweise 256.
OutputAddress	int	Logische Ausgangsadresse des verschalteten Objekts, beispielsweise 256.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Attribut	Datentyp	Beschreibung
ConnectOption	ConnectOption	Wert der bei der Verbindungsherstellung eingestellten ConnectOption: • Standardeinstellung Es können nur Module ausgewählt werden, die als gültige Verbindungspartner erkannt werden. • AllowAllModules Diese Option entspricht dem Markieren von "Zeige alle Module" in der Anwenderschnittstelle.
Channel	HW.Channel	Verschalteter Kanal
Telegram	MC.Drives.Telegram	Verschaltetes Telegramm des Antriebsobjekts
PathToDBMember	string	Verschaltete Technologieobjekt-Datenbausteinvariable
OutputTag	SW.Tags.PlcTag	Verschaltete CPU-Variable (analoge Verschaltung)
SensorIndexInActorTelegram	int	Verschaltete Sensorschnittstelle in Aktortelegramm Das Attribut ist nur für Sensorschnittstellen relevant. 0: Geber ist nicht verschaltet 1: Geber ist mit der ersten Sensorschnittstelle im Telegramm verschaltet 2: Geber ist mit der zweiten Sensorschnittstelle im Telegramm verschaltet Der Wert der Aktorschnittstelle ist stets 0.

Programmcode zur Verschaltung eines gemischtes Moduls mit Eingangs- und Ausgangsadressen am gleichen Modul

Ändern Sie den folgenden Programmcode, um ein Antriebsobjekt Servo aus der GSDML-Datei des Antriebs mit Eingangs- und Ausgangsadressen am gleichen Modul zu verschalten:

```
public void Connect_ModuleInOut()
{
    var deviceItem = GsdmlDevice.DeviceItems.First(x => x.Name == "SIEMENS telegram 105,
PZD-10/10");
    var syncAxis =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find("SynchronousAxis_1");
    syncAxis
        .GetService<AxisHardwareConnectionProvider>()
        .ActorInterface
        .Connect(deviceItem);
}
```

Programmcode zur Verschaltung von Eingangs- und Ausgangsadressen an getrennten Modulen

Ändern Sie den folgenden Programmcode, um ein Antriebsobjekt Servo aus den GSDML-Dateien der Antriebe mit Eingangs- und Ausgangsadressen an verschiedenen Modulen zu verschalten:

```
//"Input module", "Output module" and "syncAxis" must be replaced with the names in your
project.
public void Connect_ModuleIn_ModuleOut()
{
    var deviceItemOut = Plc.DeviceItems.First(di => di.Name == "Input
module").DeviceItems[0];
    var deviceItemIn = Plc.DeviceItems.First(di => di.Name == "Output
module").DeviceItems[0];
    var syncAxis =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find("SynchronousAxis_1");
    syncAxis
        .GetService<AxisHardwareConnectionProvider>()
        .ActorInterface
        .Connect(deviceItemIn, deviceItemOut, ConnectOption.AllowAllModules);
}
```

Wenn das TIA Portal das Modul nicht als Standardtelegramm erkennt, muss die Option AllowAllModules verwendet werden.

Programmcode zur Verschaltung von Technologiemodulen

Ändern Sie folgenden Programmcode, um ein Technologiemodul zu verschalten:

```
public void Connect_Channel()
{
    var tmTimer = Plc.DeviceItems.First(di => di.Name== "TM Timer DIDQ
16x24V_1").DeviceItems[0];
    var syncAxis =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find("SynchronousAxis_1");
    syncAxis.GetService<OutputCamMeasuringInputContainer>()
        .OutputCams.Find("OutputCam_1")
        .GetService<OutputCamHardwareConnectionProvider>()
        .Connect(tmTimer.Channels[0]);
}
```

Programmcode zur Verschaltung von über Startdrive ab V19 projektierten Antrieben

Startdrive ab V19 muss installiert sein.

Um ein Antriebsobjekt Servo aus Startdrive mittels Telegramm zu verschalten, ändern Sie folgenden Programmcode:

```
public void Connect_To_Startdrive(PlcSoftware plcsoftware)
{
    var mainTelegram = StartdriveDevice.DeviceItems.First(di => di.PositionNumber == 0)
        .GetService<DriveObjectContainer>().DriveObjects[0].Telegrams.Find(TelegramType.MainTele
gram);
    var syncAxis =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find("SynchronousAxis_1");

syncAxis.GetService<AxisHardwareConnectionProvider>().ActorInterface.Connect(mainTelegram)
;
}
```

Programmcode zur Verschaltung von über Startdrive < V19 projektierten Antrieben

Startdrive muss installiert sein.

Ändern Sie folgenden Programmcode, um ein Antriebsobjekt Servo aus Startdrive zu verschalten:

```
public void Connect_To_Startdrive()
{
    var mainTelegram = StartdriveDevice.DeviceItems.First(di => di.PositionNumber == 0)
        .GetService<DriveObjectContainer>().DriveObjects[0].Telegrams.Find(TelegramType.MainTele
gram);
    var syncAxis =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find("SynchronousAxis_1");
    var inputAddress = mainTelegram.Addresses.First(x => x.IoType ==
AddressIoType.Input).StartAddress;
    var outputAddress = mainTelegram.Addresses.First(x => x.IoType ==
AddressIoType.Output).StartAddress;
    syncAxis
        .GetService<AxisHardwareConnectionProvider>()
        .ActorInterface
//Byte addresses must be multiplied by 8 to get Bit addresses
        .Connect(inputAddress * 8, outputAddress * 8, ConnectOption.Default);
}
```

Programmcode zur Verschaltung von Datenbausteinvariablen

In diesem Beispiel wird der Sensor eines externen Gebers mit dem Datenbaustein "InstDBTel83" verschaltet. InstDBTel83 enthält ein Mitglied "Telegramm" des Typs "PD_Tel83":

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie den folgenden Programmcode, um eine Datenbausteinvariable zu verschalten.

```
public void Connect_PathToDBMember()  
{  
    var extEnc =  
    plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find("ExternalEncoder_1");  
    extEnc  
        .GetService<EncoderHardwareConnectionProvider>()  
        .SensorInterface  
        .Connect("InstDBTel83.Telegram.Sensor_1");  
}
```

Telegramm 750 verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_SpeedAxis, TO_PositioningAxis oder TO_SynchronousAxis V4.0 ist im Projekt angelegt.
- Ein Antrieb, der Telegramm 750 unterstützt, ist im Projekt angelegt.

Anwendung

Wenn Telegramm 750 nach Verschalten des Antriebs und der Achse hinzugefügt wurde, ist es nötig, Telegramm 750 getrennt zu verschalten. EnableTorqueData ist automatisch auf TRUE gesetzt. Der Typ der Public API TorqueHardwareConnectionInterface bietet die folgenden Methoden, die zur Verschaltung oder Trennung von Telegramm 750 verwendet werden können:

Methode	Beschreibung
void Connect(HW.Deviceltem moduleInOut)	Verschaltet mit Eingangs- und Ausgangsadressen am gleichen Modul. Verwenden Sie diese Methode zur Verschaltung von Telegramm 750 für Antriebe, die mit Hilfe einer GSD-Datei projektiert werden.
void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut)	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen
void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut, ConnectOption connectOption)	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen unter Angabe einer zusätzlichen ConnectOption.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Methode	Beschreibung
<code>void Connect(int addressIn, int addressOut, ConnectOption connectOption)</code>	Verschaltet Bitadressen direkt. Diese Methode können Sie in allen Fällen verwenden. Wird der Antrieb mit Startdrive projektiert, muss diese Methode verwendet werden. Lesen Sie die Byteadresse des Telegramms aus der Hardwarekonfiguration aus und verwenden Sie die entsprechende Bitadresse. $\text{Bitadresse} = \text{Byteadresse} * 8$
<code>void Connect(string pathToDBMember)</code>	Verschaltet mit einer Datenbausteinvariablen. Verwenden Sie diese Methode zur Verschaltung von Telegramm 750 über einen Datenbaustein. Der Datenbaustein für die Verschaltung muss vorhanden sein.
<code>void Disconnect()</code>	Trennt eine bestehende Verschaltung.

Die `TorqueHardwareConnectionInterface` kann über die Eigenschaft `TorqueInterface` am Typ `AxisHardwareConnectionProvider` abgerufen werden. Wird die Verschaltung mit Telegramm 750 nicht unterstützt, ist der Wert der Eigenschaft "null".

Ist der Antrieb über Datenbausteinvariablen verschaltet, können Sie Telegramm 750 nicht mittels Modul verschalten. Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird. Die entsprechenden Verschaltungswerte werden nur eingestellt, wenn eine Verschaltung dieser spezifischen Art besteht:

Attribut	Datentyp	Beschreibung
<code>IsConnected</code>	bool	TRUE: Schnittstelle ist verschaltet FALSE: Schnittstelle ist nicht verschaltet
<code>InputOutputModule</code>	HW.DeviceItem	Verschaltetes Modul, das Eingangs- und Ausgangsadressen enthält
<code>InputModule</code>	HW.DeviceItem	Verschaltetes Modul, das Eingangsadressen enthält. Der Wert wird auch im Fall einer bestehenden Verschaltung mit einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
<code>OutputModule</code>	HW.DeviceItem	Verschaltetes Modul, das Ausgangsadressen enthält. Der Wert wird auch im Fall einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
<code>InputAddress</code>	int	Logische Eingangsadresse des verschalteten Objekts, beispielsweise 256
<code>OutputAddress</code>	int	Logische Ausgangsadresse des verschalteten Objekts, beispielsweise 256

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Attribut	Datentyp	Beschreibung
ConnectOption	ConnectOption	Wert der bei der Verschaltungsherstellung eingestellten ConnectOption: • Standard Es können nur Module ausgewählt werden, die als gültige Verschaltungspartner erkannt werden. • AllowAllModules Diese Option entspricht dem Markieren von "Zeige alle Module" in der Anwenderschnittstelle.
PathToDB-Member	string	Verschaltete Technologieobjekt-Datenbausteinvariable

Programmcode: Telegramm 750 verschalten

Ändern Sie den folgenden Programmcode, um ein gemischtes Modul, das Eingangs- und Ausgangsadressen enthält, zu verschalten:

```
//An instance of technology object and device item is already available in the program
before
private static void ConnectTorqueInterface(TechnologicalInstanceDB technologyObject,
DeviceItem devItem)
{
    //Retrieve service AxisHardwareConnectionProvider
    AxisHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<AxisHardwareConnectionProvider>();
    //Connect TorqueInterface with DeviceItem
    connectionProvider.TorqueInterface.Connect(devItem);
}
```

Geber verschalten**Voraussetzung**

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_ExternalEncoder ist im Projekt angelegt.
- Im Projekt wird ein Objekt bestimmt, welches PROFIdrive Telegramm 81 oder 83 bereitstellt.

Anwendung

Um ein Technologieobjekt Externer Geber mit der Geber-Hardware zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API `AxisEncoderHardwareConnectionInterface` bietet die folgenden Methoden, die zur Verschaltung oder Trennung der Sensorschnittstelle verwendet werden können.

Methode	Beschreibung
<code>void Connect(HW.Deviceltem moduleInOut)</code>	Verschaltet mit Eingangs- und Ausgangsadressen am gleichen Modul. Verwenden Sie diese Methode zur Verschaltung von Gebern, die mit Hilfe einer GSD-Datei projiziert werden.
<code>void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut)</code>	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen Verwenden Sie diese Methode zur Verschaltung von Gebern, die mit Hilfe einer GSD-Datei projiziert werden.
<code>void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut, ConnectOption connectOption)</code>	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen unter Angabe einer zusätzlichen <code>ConnectOption</code> Verwenden Sie diese Methode zur Verschaltung von Gebern, die mit Hilfe einer GSD-Datei projiziert werden.
<code>void Connect(HW.Channel channel)</code>	Verschaltet mit einem Kanal. Verwenden Sie diese Methode zur Verschaltung von Technologiemodulen.
<code>void Connect(int addressIn, int addressOut, ConnectOption connectOption)</code>	Verschaltet Bitadressen direkt. Diese Methode können Sie in allen Fällen verwenden. Wird der Geber mit Startdrive projiziert, muss diese Methode verwendet werden. Lesen Sie die Byteadresse des Telegramms aus der Hardwarekonfiguration aus und verwenden Sie die entsprechende Bitadresse. $\text{Bitadresse} = \text{Byteadresse} * 8$
<code>void Connect(string pathToDBMember)</code>	Verschaltet mit einer Datenbausteinvariablen. Verwenden Sie diese Methode zur Verschaltung eines externen Gebers über einen Datenbaustein. Der Datenbaustein für die Verschaltung muss vorhanden sein.
<code>void Connect(SW.Tags.PlcTag outputTag)</code>	Nicht relevant für das Verschalten von Gebern
<code>void Disconnect()</code>	Trennt eine bestehende Verschaltung.

Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird. Die entsprechenden Verschaltungswerte werden nur eingestellt, wenn eine Verschaltung dieser spezifischen Art besteht.

Attribut	Datentyp	Beschreibung
<code>IsConnected</code>	<code>bool</code>	TRUE: Schnittstelle ist verschaltet FALSE: Schnittstelle ist nicht verschaltet
<code>InputOutputModule</code>	<code>HW.Deviceltem</code>	Verschaltetes Modul, das Eingangs- und Ausgangsadressen enthält
<code>InputModule</code>	<code>HW.Deviceltem</code>	Verschaltetes Modul, das Eingangsadressen enthält Der Wert wird auch im Falle einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Attribut	Datentyp	Beschreibung
OutputModule	HW.DeviceItem	Verschaltetes Modul, das Ausgangsadressen enthält Der Wert wird auch im Falle einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
InputAddress	int	Logische Eingangsadresse des verschalteten Objekts, beispielsweise 256.
OutputAddress	int	Logische Ausgangsadresse des verschalteten Objekts, beispielsweise 256.
ConnectOption	ConnectOption	Wert der bei der Verbindungsherstellung eingestellten ConnectOption: • Default Es können nur Module ausgewählt werden, die als gültige Verbindungspartner erkannt werden. • AllowAllModules Diese Option entspricht dem Markieren von "Zeige alle Module" in der Anwenderschnittstelle.
Channel	HW.Channel	Verschalteter Kanal
PathToDBMember	string	Verschaltete Datenbausteinvariable
OutputTag	SW.Tags.PlcTag	Nicht relevant für das Verschalten von Gebern
SensorIndexInActorTelegram	int	Verschaltetes Sensortelegamm Das Attribut ist nur für Sensorschnittstellen relevant. 0: Geber ist nicht verschaltet 1: Geber ist mit der ersten Sensorschnittstelle im Telegramm verschaltet 2: Geber ist mit der zweiten Sensorschnittstelle im Telegramm verschaltet Der Wert der Aktorschnittstelle ist stets 0.

Programmcode: Geber verschalten

Ändern Sie den folgenden Programmcode, um ein Technologieobjekt Externer Geber zu verschalten:

```
//An instance of technology object and device item is already available in the program
before
private static void UseServiceEncoderHardwareConnectionProvider(TechnologicalInstanceDB
technologyObject, DeviceItem devItem)
{
    //Retrieve service EncoderHardwareConnectionProvider
    EncoderHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<EncoderHardwareConnectionProvider>();

    //Connect SensorInterface with DeviceItem
    connectionProvider.SensorInterface.Connect(devItem);

    //Check ConnectionState of SensorInterface
    bool sensorInterfaceConnectionState = connectionProvider.SensorInterface.IsConnected;
}
```


Nocken und Nockenspur mit Hardware verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_OutputCam oder TO_CamTrack ist im Projekt angelegt.
- Ein Digitalausgabemodul ist im Projekt angelegt, zum Beispiel TM Timer DIDQ.

Anwendung

Um ein Technologieobjekt Nocken oder Nockenspur mit einem Digitalausgang zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API OutputCamHardwareConnectionProvider bietet die folgenden Methoden, die zur Verschaltung oder Trennung der Aktor- oder Sensorschnittstellen verwendet werden können:

Methode	Beschreibung
void Connect(HW.Channel channel)	Verschaltet mit einem Kanal
void Connect(SW.Tags.PlcTag outputTag)	Verschaltet mit einer CPU-Variablen
void Connect(int address)	Verschaltet Bit-Adressen direkt
void Disconnect()	Trennt eine existierende Verschaltung

Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird:

Attribut	Datentyp	Beschreibung
IsConnected	bool	TRUE: Technologieobjekt ist verschaltet FALSE: Technologieobjekt ist nicht verschaltet
Channel	HW.Channel	Verschalteter Kanal
OutputTag	SW.Tags.PlcTag	Verschaltete CPU-Variable
OutputAddress	int	Logische Ausgangsadresse des verschalteten Objekts, beispielsweise 256.

Programmcode: Technologieobjekt Nocken oder Nockenspur verschalten

Ändern Sie den folgenden Programmcode, um ein Technologieobjekt Nocken oder Nockenspur zu verschalten:

```
//An instance of technology object and channel item is already available in the program
before
private static void UseServiceOutputCamHardwareConnectionProvider(TechnologicalInstanceDB
technologyObject, Channel channel)

{
    //Retrieve service OutputCamHardwareConnectionProvider
    OutputCamHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<OutputCamHardwareConnectionProvider>();

    //Connect technology object with Channel
    connectionProvider.Connect(channel);

    //Check ConnectionState of technology object
    bool connectionState = connectionProvider.IsConnected;
}
```

Programmcode zur Verschaltung von PLC-Variablen

Ändern Sie den folgenden Programmcode, um einen Nocken mit einer PLC-Variablen zu verschalten. Die PLC-Variable zeigt auf die Ausgangsadresse:

```
public void Connect_PlcTag(PlcSoftware plcSoftware)
{
    var outputCamTag = plcSoftware.TagTableGroup.TagTables.Find("My Tag
Table").Tags.Find("OutputCamTag");
    var syncAxis =
    plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find("SynchronousAxis_1");
    syncAxis.GetService<OutputCamMeasuringInputContainer>()
    .OutputCams.Find("OutputCam_1")
    .GetService<OutputCamHardwareConnectionProvider>()
    .Connect(outputCamTag);
}
```

Messtaster mit Hardware verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500 CPU ist im Projekt angelegt.

- Ein Technologieobjekt des Typs TO_MeasuringInput ist im Projekt angelegt.
- Ein Digitaleingabemodul ist am Antrieb oder im Projekt angelegt, zum Beispiel TM Timer DIDQ.

Anwendung

Um ein Technologieobjekt Messtaster mit einem Digitaleingang zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API MeasuringInputHardwareConnectionProvider bietet die folgenden Methoden, die zur Verschaltung oder Trennung der Aktor- oder Sensorschnittstellen verwendet werden können:

Methode	Beschreibung
void Connect(HW.Channel channel)	Verschaltet mit einem Kanal
void Connect(HW.DeviceItem moduleIn, int channelIndex)	Verschaltet mit einem Modul und legt einen zusätzlichen Kanalindex fest
void Connect(int address)	Verschaltet Bit-Adressen direkt • Bei TM-Kanälen können die Bitadressen über die Formel $\text{bit\_address} = \text{module\_start\_address} * 8 + \text{channel\_index}$ berechnet werden. • Bei Telegramm 39x ist die zugehörige Formel $\text{bit\_address} = \text{module\_start\_address} * 8 + 24 + \text{number\_of\_measuringinput\_in\_telegram}$.
void Disconnect()	Trennt eine existierende Verschaltung

Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird:

Attribut	Datentyp	Beschreibung
IsConnected	bool	TRUE: Technologieobjekt ist verschaltet FALSE: Technologieobjekt ist nicht verschaltet
InputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangsadressen enthält
ChannelIndex	int	Index des verschalteten Kanals hinsichtlich des Eingabemoduls
Channel	HW.Channel	Verschalteter Kanal
InputAddress	int	Die logische Eingangsadresse des verschalteten Objekts ist beispielsweise 256.

Programmcode: Technologieobjekt Messtaster verschalten

Ändern Sie den folgenden Programmcode, um ein Technologieobjekt Messtaster zu verschalten:

```
//An instance of technology object and channel item is already available in the program
before
private static void
UseServiceMeasuringInputHardwareConnectionProvider(TechnologicalInstanceDB
technologyObject, Channel channel)
{
    //Retrieve service MeasuringInputHardwareConnectionProvider
    MeasuringInputHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<MeasuringInputHardwareConnectionProvider>();

    //Connect technology object with Channel
    connectionProvider.Connect(channel);

    //Check ConnectionState of technology object
    bool connectionState = connectionProvider.IsConnected;
}
```

Gleichlaufachse mit Leitwerten verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_PositioningAxis, TO_SynchronousAxis, TO_ExternalEncoder oder TO_LeadingAxisProxy als Leitachse ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_SynchronousAxis als Folgeachse ist im Projekt angelegt.

Anwendung

Um ein Technologieobjekt Gleichlaufachse mit Leitwerten zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API SynchronousAxisMasterValues bietet die folgenden Methoden, die zur Verschaltung oder Trennung von Leitwerten verwendet werden können. Leitwerte können als Sollwertkopplungen (S7-1500 CPU, S7-1500T CPU) oder Istwertkopplungen (S7-1500T CPU) verschaltet werden. Alle Methoden und Attribute sind für beide Arten von Kopplungen relevant.

Methode	Beschreibung
int IndexOf (TechnologicalInstanceDB element)	Gibt den entsprechenden Index eines Leitwertes zurück
bool Contains (TechnologicalInstanceDB element)	TRUE: Der Behälter enthält den Leitwert FALSE: Der Behälter enthält nicht den Leitwert
IEnumerator GetEnumerator <TechnologicalInstanceDB>()	Dient zur Unterstützung jeder Iteration

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Methode	Beschreibung
void Add (TechnologicalInstanceDB element)	Verschaltet die Folgeachse mit einem Leitwert
bool Remove (TechnologicalInstanceDB element)	Trennt die Folgeachse vom Leitwert TRUE: Verschaltung wurde erfolgreich getrennt FALSE: Verschaltung konnte nicht getrennt werden

Sie können die folgenden schreibgeschützten Attribute verwenden:

Attribut	Datentyp	Beschreibung
Count	int	Anzahl der Leitwerte
IsReadOnly	bool	TRUE: Der Behälter ist schreibgeschützt FALSE: Der Behälter ist nicht schreibgeschützt
Parent	IEngineeringObject	Gibt die übergeordneten Werte des Behälters zurück. In diesem Fall ist der übergeordnete Wert der Service SynchronousAxis-MasterValues.
this [id] { get; }	TechnologicalInstanceDB	Indexbasierter Zugriff auf Leitwerte

Sie können die folgenden Attribute verwenden, um eine Gleichlaufachse mit einem Leitwert zu verschalten:

Attribut	Datentyp	Beschreibung
SetPointCoupling	SW.TechnologicalObjects.TechnologicalInstanceDBAssociation	Das Attribut erhält einen Container der verschalteten Leitachse mit Sollwertkopplung.
ActualValueCoupling	SW.TechnologicalObjects.TechnologicalInstanceDBAssociation	Das Attribut erhält einen Container der verschalteten Leitachse mit Istwertkopplung.
DelayedCoupling	SW.TechnologicalObjects.TechnologicalInstanceDBAssociation	Das Attribut enthält alle Masterwerte, die über verzögerte Werte gekoppelt sind.

Programmcode: Gleichlaufachse mit einem Leitwert verschalten

Ändern Sie den folgenden Programmcode, um eine Gleichlaufachse mit einem Leitwert zu verschalten:

```
//An instance of leading axis and following axis is already available in the program before
private static void UseServiceSynchronousAxisMasterValues(TechnologicalInstanceDB
masterTechnologyObject, TechnologicalInstanceDB synchronousTechnologyObject)
{
    //Retrieve service SynchronousAxisMasterValues
    SynchronousAxisMasterValues masterValues =
    synchronousTechnologyObject.GetService<SynchronousAxisMasterValues>();

    //Connect following axis and leading axis with setpoint coupling
    masterValues.SetPointCoupling.Add(masterTechnologyObject);

    //Get container of connected leading axis with setpoint coupling
    TechnologicalInstanceDBAssociation setPointMasterValues =
    masterValues.SetPointCoupling;

    //Remove connected leading axis with setpoint coupling
    masterValues.SetPointCoupling.Remove(masterTechnologyObject);

    //Connect following axis and leading axis with actual value coupling
    masterValues.ActualValueCoupling.Add(masterTechnologyObject);

    //Get container of connected leading axis with actual value coupling
    TechnologicalInstanceDBAssociation actualValueMasterValues =
    masterValues.ActualValueCoupling;

    //Remove connected leading axis with actual value coupling
    masterValues.ActualValueCoupling.Remove(masterTechnologyObject);
}
```

Kinematik mit Leitwerten für Bandverfolgung (Conveyor tracking) verschalten**Voraussetzung**

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500T PLC ist im Projekt angelegt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277).
- Ein Technologieobjekt des Typs TO_PositioningAxis, TO_SynchronousAxis oder TO_ExternalEncoder als Leitachse ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_Kinematics als Folgeachse ist im Projekt angelegt.

Verwendung

Um Leitwerte für die Bandverfolgung zu konfigurieren, verwenden Sie den Service `SW.TechnologicalObjects.Motion.ConveyorTrackingLeadingValues` für das TO Kinematik $\geq$ V5.0.

Methode	Beschreibung
<code>int IndexOf (TechnologicalInstanceDB element)</code>	Gibt den entsprechenden Index eines Leitwertes zurück
<code>bool Contains (TechnologicalInstanceDB element)</code>	TRUE: Der Behälter enthält den Leitwert FALSE: Der Behälter enthält nicht den Leitwert
<code>IEnumerator GetEnumerator <TechnologicalInstanceDB>()</code>	Dient zur Unterstützung jeder Iteration
<code>void Add (TechnologicalInstanceDB element)</code>	Verschaltet die Folgeachse mit einem Leitwert
<code>bool Remove (TechnologicalInstanceDB element)</code>	Trennt die Folgeachse vom Leitwert TRUE: Verschaltung wurde erfolgreich getrennt FALSE: Verschaltung konnte nicht getrennt werden

Sie können die folgenden schreibgeschützten Attribute verwenden:

Attribut	Datentyp	Beschreibung
Count	int	Anzahl der Leitwerte
IsReadOnly	bool	TRUE: Der Behälter ist schreibgeschützt FALSE: Der Behälter ist nicht schreibgeschützt
Parent	IEngineeringObject	Gibt die übergeordneten Werte des Behälters zurück. In diesem Fall ist der übergeordnete Wert der Service <code>SynchronousAxisMasterValues</code> .
<code>this [id] { get; }</code>	TechnologicalInstanceDB	Indexbasierter Zugriff auf Leitwerte

Sie können die folgenden Attribute verwenden, um eine Kinematikachse mit einem Leitwert zu verschalten:

Attribut	Datentyp	Beschreibung
SetPointCoupling	<code>SW.TechnologicalObjects.TechnologicalInstanceDBAssociation</code>	Das Attribut erhält einen Container der verschalteten Leitachse mit Sollwertkopplung.
ActualValueCoupling	<code>SW.TechnologicalObjects.TechnologicalInstanceDBAssociation</code>	Das Attribut erhält einen Container der verschalteten Leitachse mit Istwertkopplung.
DelayedCoupling	<code>SW.TechnologicalObjects.TechnologicalInstanceDBAssociation</code>	Das Attribut enthält alle Masterwerte, die über verzögerte Werte gekoppelt sind.

Bandverfolgungsachse der Kinematik mit einem Leitwert verschalten

Ein Beispiel für die Verschaltung ist im Thema Gleichlaufachse mit Leitwerten verschalten (Seite 592) enthalten.

Exportieren und Importieren des Technologieobjekts Cam (S7-1500T)

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500 CPU ist im Projekt angelegt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)
- Das Technologieobjekt liegt vor.

Anwendung

Um die Daten eines Technologieobjekts Kurvenscheibe vom Typ TO_Cam oder TO_Cam_10k zu exportieren oder zu importieren, müssen Sie das Format und das gewünschte Trennzeichen angeben. Der Typ der Public API CamDataSupport bietet die folgenden Methoden, die zum Exportieren der Daten des Technologieobjekts Kurvenscheibe verwendet werden können: Weitere Informationen zum Exportformat von Kurvenscheiben finden Sie im Funktionshandbuch S7-1500/S7-1500T zu Gleichlaufunktionen im Kapitel zum Importieren/Exportieren von Kurvenscheiben.

Methode	Beschreibung
void SaveCamDataBinary(System.IO.FileInfo destinationFile)	Speichert die Kurvenscheibendaten im Binärformat in der Zielfeile.
void SaveCamDataPointList(System.IO.FileInfo destinationFile, CamDataFormatSeparator separator, int samplePoints)	Speichert die Kurvenscheibendaten im Format "Punktliste" in der Zielfeile.
void SaveCamData(System.IO.FileInfo destinationFile, CamDataFormat format, CamDataFormatSeparator separator)	Speichert die Kurvenscheibendaten in der Zielfeile. Sie können das Datenformat als "MCD", "SCOUT" oder "Pointlist" und als Trennzeichen "Tabulator" oder "Komma" angeben. Wenn Sie "PointList" wählen, werden 360 Interpolationspunkte exportiert.
void LoadCamData(System.IO.FileInfo sourceFile, CamDataFormatSeparator separator)	Lädt die Kurvenscheibendaten im Format "MCD", "SCOUT" oder "Pointlist" in das Projekt.
void LoadCamDataBinary(System.IO.FileInfo sourceFile)	Lädt die Kurvenscheibendaten aus einer Binärdatei in das Projekt.

Die folgenden Attribute können Sie verwenden:

Attribut	Datentyp	Beschreibung
separator	CamDataFormatSeparator	Zulässige Werte • Tabulator • Komma
samplePoints	int	Anzahl zu exportierender Interpolationspunkte.

Attribut	Datentyp	Beschreibung
format	CamDataFormat	Zulässige Werte • MCD • SCOUT • Punkteliste
destinationFile	System.IO.FileInfo	Name der Zieldatei. Darf nicht null sein. Zugriffsrechte und ausreichend Speicherplatz müssen auf dem Speichermedium gegeben sein. Eine vorhandene Datei wird überschrieben.
sourceFile	System.IO.FileInfo	Name der Quelldatei. Darf nicht null sein. Zugriffsrechte müssen gegeben sein. Der Inhalt muss im angegebenen Format vorliegen.

Programmcode: Kurvenscheibendaten speichern

Um Kurvenscheibendaten zu speichern, ändern Sie folgenden Programmcode:

```
//An instance of technology object is already available in the program before
private static void ExportCamData(TechnologicalInstanceDB technologyObject,
System.IO.FileInfo destinationFile)
{
    //Retrieve service CamDataSupport
    CamDataSupport camData = technologyObject.GetService<CamDataSupport>();

    //Save cam data in MCD format, using the separator Tab
    camData.SaveCamData(destinationFile, CamDataFormat.MCD, CamDataFormatSeparator.Tab);
}
```

Programmcode: Kurvenscheibendaten laden

Um Kurvenscheibendaten zu laden, ändern Sie folgenden Programmcode:

```
//An instance of technology object is already available in the program before
private static void ImportCamData(TechnologicalInstanceDB technologyObject,
System.IO.FileInfo sourceFile)
{
    //Retrieve service CamDataSupport
    CamDataSupport camData = technologyObject.GetService<CamDataSupport>();

    //Load cam data from source file, using the separator Tab
    camData.LoadCamData(sourceFile, CamDataFormatSeparator.Tab);
}
```

Quellcode eines Interpreter-Programms (S71500T) laden und speichern

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Eine S7-1500T PLC ist im Projekt angelegt.
Siehe PLC-Target und HMI-Target abfragen (Seite 277).
- Ein Technologieobjekt des Typs TO_InterpreterProgram ist im Projekt angelegt.

Verwendung

Der Typ der Public API InterpreterProgramSupport bietet die folgenden Methoden, die zum Laden des MCL-Programms aus einer Textdatei in ein Technologieobjekt TO_InterpreterProgram oder zum Speichern des Programms in einer Textdatei eingesetzt werden können.

Das MCL-Programm wird von einem Pfad zur Quelldatei angegeben.

Methode	Beschreibung
void SaveSource(System.IO.FileInfo fileInfo)	Speichert das MCL-Programm aus dem Technologieobjekt TO_InterpreterProgram in der angegebenen Textdatei.
void LoadSource(System.IO.FileInfo fileInfo)	Lädt die angegebene Textdatei mit dem MCL-Programm in das Technologieobjekt TO_InterpreterProgram.

Programmcode zum Speichern eines MCL-Programms

Ändern Sie folgenden Programmcode, um ein Technologiemodul zu verschalten:

```
//An instance of technology object is already available in the program before
private static void ExportIprProgramSourceCode(TechnologicalInstanceDB technologyObject,
System.IO.FileInfo destinationFile)
{
    //Retrieve service InterpreterProgramSupport
    InterpreterProgramSupport iprProgramSourceCode =
technologyObject.GetService<InterpreterProgramSupport>();

    //Save source code in the destination file
    iprProgramSourceCode.SaveSource(destinationFile);
}
```

Programmcode zum Laden eines MCL-Programms

Ändern Sie folgenden Programmcode, um ein Technologiemodul zu verschalten:

```
//An instance of technology object is already available in the program before
private static void ImportIprProgramSourceCode(TechnologicalInstanceDB technologyObject,
System.IO.FileInfo sourceFile)
{
    //Retrieve service InterpreterProgramSupport
    InterpreterProgramSupport iprProgramSourceCode =
    technologyObject.GetService<InterpreterProgramSupport>();

    //Load source code from source file
    iprProgramSourceCode.LoadSource(sourceFile);
}
```

5.11.4.19 PID-Regelung

Parameter für PID_Compact, PID_3Step, PID_Temp, CONT_C, CONT_S, TCONT_CP und TCONT_S

Eine Liste mit allen verfügbaren Variablen finden Sie im Funktionshandbuch SIMATIC S7-1200/ S7-1500 PID control im Internet (<https://support.industry.siemens.com/cs/ww/de/view/108210036>).

Hinweis

Im TIA Portal finden Sie in der Parametersicht der Technologieobjektkonfiguration die Spalte "Name in Openness".

5.11.4.20 Zählen

Parameter für High_Speed_Counter und SSI_Absolute_Encoder

Eine Liste sämtlicher verfügbarer Parameter finden Sie in der Produktinformation "Parameter von Technologieobjekten in TIA Portal Openness" im Internet (<https://support.industry.siemens.com/cs/ww/de/view/109744932>).

Für jeden Parameter werden die folgenden Eigenschaften angegeben:

- Name in der Konfiguration (TIA Portal)
- Name in Openness
- Datentyp in Openness
- Standardzugriff
- Wertebereich

Weitere Informationen

Weitere Informationen finden Sie im Funktionshandbuch SIMATIC S7-1500, ET 200MP, ET 200SP Zählen, Messen und Positionserfassung im Internet (<http://support.automation.siemens.com/WW/view/de/59709820>).

5.11.4.21 Easy Motion Control

Parameter für AXIS_REF

Eine Liste sämtlicher verfügbarer Parameter finden Sie in der Produktinformation "Parameter von Technologieobjekten in TIA Portal Openness" im Internet (<https://support.industry.siemens.com/cs/ww/de/view/109744932>).

Für jeden Parameter werden die folgenden Eigenschaften angegeben:

- Name in der Konfiguration (TIA Portal)
- Name in Openness
- Datentyp in Openness
- Standardzugriff
- Wertebereich

Hinweis

Im TIA Portal finden Sie in der Parametersicht der Technologieobjektkonfiguration die Spalte "Name in Openness".

Weitere Informationen

Weitere Informationen zu Easy Motion Control finden Sie im Informationssystem von STEP 7 (TIA Portal).

5.11.5 Variablen und Variablentabellen

5.11.5.1 Starten des PLC-Variableneditors

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine Instanz des TIA Portals ist mit Benutzeroberfläche geöffnet.

Programmcode

Ändern Sie den folgenden Programmcode, um den entsprechenden Editor für eine Objektreferenz des Typs `PlcTagTable` in der Instanz des TIA Portals zu starten:

```
//Öffnet Variablentabelle im Editor "Tags"
private static void OpenTagtableInEditor(PlcSoftware plcSoftware)
{
    PlcTagTable plcTagTable = plcSoftware.TagTableGroup.TagTables.Find("MyTagTable");
    plcTagTable.ShowInEditor();
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

5.11.5.2 Systemgruppen für PLC-Variablen abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine `PlcSoftware`-Instanz wurde von einem PLC-Geräteelement abgerufen.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)

Programmcode

Ändern Sie den folgenden Programmcode, um eine Systemgruppe für PLC-Variablen abzufragen:

```
//Retrieves the plc tag table group from a plc
private PlcTagTableSystemGroup GetControllerTagfolder(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    return plcTagTableSystemGroup;
}
```

5.11.5.3 PLC-Variablentabelle anlegen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PlcSoftware-Instanz wurde von einem PLC-Geräteelement abgerufen.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)

Programmcode

Um die PLC-Variablentabelle anzulegen, ändern Sie folgenden Programmcode: Es wird eine neue Variablentabelle mit dem angegebenen Namen in der Zusammensetzung angelegt.

```
PlcTagTable myTable = plc.TagTableGroup.TagTables.Create("myTable");
```

Siehe auch

PLC-Target und HMI-Target abfragen (Seite 277)

5.11.5.4 Benutzerdefinierte Gruppen für PLC-Variablen enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Eine PlcSoftware-Instanz wurde von einem PLC-Geräteelement abgerufen.
Siehe PLC-Target und HMI-Target abfragen (Seite 277)

Verwendung

Enthaltene Unterordner werden beim Enumerieren rekursiv berücksichtigt.

Programmcode: Benutzerdefinierte Gruppen für PLC-Variablen enumerieren

Ändern Sie den folgenden Programmcode, um benutzerdefinierte Gruppen für PLC-Variablen zu enumerieren:

```
//Enumerates all plc tag table user groups including subgroups
private static void EnumeratePlcTagTableUserGroups(PlcSoftware plcSoftware)
{
    foreach (PlcTagTableUserGroup plcTagTableUsergroup in plcSoftware.TagTableGroup.Groups)
    {
        EnumerateTagTableUserGroups(plcTagTableUsergroup);
    }
}
private static void EnumerateTagTableUserGroups(PlcTagTableUserGroup tagTableUsergroup)
{
    foreach (PlcTagTableUserGroup plcTagTableUsergroup in tagTableUsergroup.Groups)
    {
        EnumerateTagTableUserGroups(plcTagTableUsergroup);
        // recursion
    }
}
```

Programmcode: Auf eine benutzerdefinierte Gruppe zugreifen

Ändern Sie den folgenden Programmcode, um auf eine benutzerdefinierte Gruppe für PLC-Variablen zuzugreifen:

```
//Gives individual access to a specific plc tag table user folder
private static void AccessPlcTagTableUserGroupWithFind(PlcSoftware plcSoftware, string
folderToFind)
{
    PlcTagTableUserGroupComposition plcTagTableUserGroupComposition =
plcSoftware.TagTableGroup.Groups;
    PlcTagTableUserGroup controllerTagUserFolder =
plcTagTableUserGroupComposition.Find(folderToFind);
    // The parameter specifies the name of the user folder
}
```

5.11.5.5 Benutzerdefinierte Gruppen für PLC-Variablen erzeugen**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt die Erzeugung einer benutzerdefinierten Gruppe für PLC-Variablen.

Programmcode

Ändern Sie den folgenden Programmcode, um eine benutzerdefinierte Gruppe für PLC-Variablen zu erzeugen:

```
//Creates a plc tag table user group
private static void CreatePlcTagTableUserGroup(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup systemGroup = plcSoftware.TagTableGroup;
    PlcTagTableUserGroupComposition groupComposition = systemGroup.Groups;
    PlcTagTableUserGroup myCreatedGroup = groupComposition.Create("MySubGroupName");
    // Optional;
    // create a subgroup
    PlcTagTableUserGroup mySubCreatedGroup =
myCreatedGroup.Groups.Create("MySubSubGroupName");
}
```

5.11.5.6 Benutzerdefinierte Gruppen für PLC-Variablen löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt das Löschen einer bestimmten benutzerdefinierten Gruppe für PLC-Variablentabellen.

Programmcode

Ändern Sie den folgenden Programmcode, um eine bestimmte benutzerdefinierte Gruppe für PLC-Variablentabellen zu löschen:

```
private static void DeletePlcTagTableUserGroup(PlcSoftware plcSoftware)
{
    PlcTagTableUserGroup group = plcSoftware.TagTableGroup.Groups.Find("MySubGroupName");
    if (group != null)
    {
        group.Delete();
    }
}
```

5.11.5.7 PLC-Variablentabellen in einem Ordner enumerieren**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode: PLC-Variablentabellen enumerieren

Ändern Sie den folgenden Programmcode, um alle PLC-Variablentabellen in Systemgruppen oder benutzerdefinierten Gruppen zu enumerieren:

```
//Enumerates all plc tag tables in a specific system group or and user group
private static void EnumerateAllPlcTagTablesInFolder(PlcSoftware plcSoftware)
{
    PlcTagTableComposition tagTables = plcSoftware.TagTableGroup.TagTables;
    // alternatively, PlcTagTableComposition tagTables =
    plcSoftware.TagTableGroup.Groups.Find("UserGroup XYZ").TagTables;
    foreach (PlcTagTable tagTable in tagTables)
    {
        // add code here
    }
}
```

Programmcode: Auf PLC-Variablentabellen zugreifen

Um auf die PLC-Variablentabelle zuzugreifen, ändern Sie folgenden Programmcode:

```
//Gives individual access to a specific Plc tag table
private static void AccessToPlcTagTableWithFind(PlcSoftware plcSoftware)
{
    PlcTagTableComposition tagTables = plcSoftware.TagTableGroup.TagTables;
    // alternatively, PlcTagTableComposition tagTables =
    plcSoftware.TagTableGroup.Groups.Find("UserGroup XYZ").TagTables;
    PlcTagTable controllerTagTable = tagTables.Find("Tag table XYZ");
    // The parameter specifies the name of the tag table
}
```

5.11.5.8 Informationen einer PLC-Variablentabelle abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Über PLC-Variablentabellen können Sie auf Anwenderkonstanten, Systemkonstanten und Variablen zugreifen. Die Anzahl der Variablenzusammensetzung einer Variablentabelle entspricht der Anzahl der Variablen in der Variablentabelle. Die `PLCTagTable` enthält die folgenden Navigatoren, Attribute und Aktionen.

In der PLC-Variablentabelle wird auf die folgenden Attribute zugegriffen.

Name	Typ	Typ
IsDefault	Bool	Schreibgeschützt
ModifiedTimeStamp	DateTime	Schreibgeschützt
Name	String	Schreibgeschützt

Die PLC-Variablentabelle enthält die nachstehend aufgeführten Aktionen.

Name	Rückgabotyp	Beschreibung
Delete	leer	Löscht die Instanz. Löst eine Ausnahme aus, wenn IsDefault wahr ist.
Exportieren	leer	Exportiert SIMATIC ML einer PLC-Variablentabelle.
ShowInEditor	leer	Zeigt die Variablentabelle im PLC-Variablentabelleneditor an.

Programmcode

Um die Informationen einer PLC-Variablentabelle abzufragen, ändern Sie den folgenden Programmcode:

```
private static void AccessPlcConstantsUsingFind(PlcTagTable tagTable)
{
    PlcUserConstantComposition plcUserConstants = tagTable.UserConstants;
    PlcUserConstant plcUserConstant = plcUserConstants.Find("Constant XYZ");
    //PlcSystemConstantComposition plcSystemConstants = tagTable.SystemConstants;
    //PlcSystemConstant plcSystemConstant = plcSystemConstants.Find("Constant XYZ");
}
private static void EnumeratePlcTags(PlcTagTable tagTable)
{
    PlcTagComposition plcTags = tagTable.Tags;
    foreach (PlcTag plcTag in plcTags)
    {
        string name = plcTag.Name; string typeName = plcTag.DataTypeName;
        string logicalAddress = plcTag.LogicalAddress;
    }
}
private static void EnumeratePlcTagsUsingFind(PlcTagTable tagTable)
{
    PlcTagComposition plcTags = tagTable.Tags;
    PlcTag plcTag = plcTags.Find("Constant XYZ");
}
```

5.11.5.9 Zeitpunkt der letzten Änderung einer PLC-Variablentabelle lesen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Das Format des Zeitstempels ist UTC.

Programmcode

Um den Zeitstempel einer bestimmten PLC-Variablentabelle zu lesen, ändern Sie den folgenden Programmcode:

```
//Reads Time-Stamp of a plc Tag Table
private static void GetLastModificationDateOfTagtable(PlcSoftware plcSoftware)
{
    PlcTagTable plcTagTable = plcSoftware.TagTableGroup.TagTables.Find("MyTagTable");
    DateTime modifiedTagTableTimeStamp = plcTagTable.ModifiedTimeStamp;
}
```

5.11.5.10 PLC-Variablentabelle aus einer Gruppe löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie den folgenden Programmcode, um eine bestimmte Variablentabelle aus einer Gruppe zu löschen:

```
//Deletes a PlcTagTable of a group
private static void DeletePlcTagTableInAGroup(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup group = plcSoftware.TagTableGroup;
    PlcTagTable tagtable = group.TagTables.Find("MyTagTable");
    if (tagtable != null)
    {
        tagtable.Delete();
    }
}
```

5.11.5.11 PLC-Variablen enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode: PLC-Variablen in Variablentabellen enumerieren

Um alle PLC-Variablen in einer Variablentabelle zu enumerieren, ändern Sie den folgenden Programmcode:

```
//Enumerates all plc tags in a specific tag table
private static void EnumerateAllPlcTagsInTagTable(PlcSoftware plcSoftware)
{
    PlcTagTable tagTable = plcSoftware.TagTableGroup.TagTables.Find("Tagtable XYZ");
    foreach (PlcTag tag in tagTable.Tags)
    {
        // add code here
    }
}
```

5.11.5.12 Auf PLC-Variablen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Der Typ `PlcTagComposition` stellt eine Sammlung von PLC-Variablen dar.

Programmcode: Auf eine bestimmte PLC-Variable zugreifen

Um auf die gewünschte PLC-Variable zuzugreifen, ändern Sie folgenden Programmcode. Sie haben Zugriff auf die folgenden Attribute:

- Name (nur Lesezugriff)
- Name des Datentyps
- Logische Adresse
- Kommentar

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

- ExternalAccessible
- ExternalVisible
- ExternalWritable
- IsSafety (nur Lesezugriff)

```
//Gives individual access to a specific plc tag
private static void AccessPlcTag(PlcTagTable tagTable)
{
    PlcTag tag = tagTable.Tags.Find("Tag XYZ");
    // The parameter specifies the name of the tag
}
```

Programmcode: Variablen anlegen

Ändern Sie folgenden Programmcode:

```
private static void CreateTagInPLCTagtable(PlcSoftware plcsoftware)
// Create a tag in a tag table with default attributes
{
    string tagName = "MyTag";
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcTagComposition tagComposition = table.Tags;
    PlcTag tag = tagComposition.Create(tagName);
}
```

Ändern Sie folgenden Programmcode:

```
private static void CreateTagInPLCTagtable(PlcSoftware plcsoftware)
// Create a tag of data type bool and logical address not set
{
    string tagName = "MyTag";
    string dataType = "Bool";
    string logicalAddress = "";
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcTagComposition tagComposition = table.Tags;
    PlcTag tag = tagComposition.Create(tagName, dataType, logicalAddress);
}
```

Programmcode: Variablen löschen

Ändern Sie folgenden Programmcode:

```
private static void DeleteTagFromPLCTagtable(PlcSoftware plcsoftware)
// Deletes a single tag of a tag table
{
    string tagName = "MyTag";
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcTagComposition tagComposition = table.Tags;
    PlcTag tag = tagComposition.Find(tagName);
    if (tag != null)
    {
        tag.Delete();
    }
}
```

5.11.5.13 Auf PLC-Konstanten zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Der Typ `PlcUserConstantComposition` stellt eine Sammlung von PLC-Benutzerkonstanten dar. Sie haben Zugriff auf die folgenden Attribute:

- Name (nur Lesezugriff)
- Name des Datentyps
- Wert

Der Typ `PlcSystemConstantComposition` stellt eine Sammlung von PLC-Systemkonstanten dar. Sie haben Zugriff auf die folgenden Attribute:

- Name (nur Lesezugriff)
- Datentypname (nur Lesezugriff)
- Wert (nur Lesezugriff)

Programmcode: Benutzerkonstanten anlegen

Ändern Sie folgenden Programmcode:

```
private static void CreateUserConstantInPLCTagtable(PlcSoftware plcsoftware)
// Create a user constant in a tag table
{
    string constantName = "MyConstant";
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcUserConstantComposition userConstantComposition = table.UserConstants;
    PlcUserConstant userConstant = userConstantComposition.Create(constantName);
}
```

Programmcode: Benutzerkonstanten löschen

Ändern Sie folgenden Programmcode:

```
private static void DeleteUserConstantFromPLCTagtable(PlcSoftware plcsoftware)
// Deletes a single user constant of a tag table
{
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcUserConstantComposition userConstantComposition = table.UserConstants;
    PlcUserConstant userConstant = userConstantComposition.Find("MyConstant");
    if (userConstant != null)
    {
        userConstant.Delete();
    }
}
```

Programmcode: Auf Systemkonstanten zugreifen

Ändern Sie folgenden Programmcode:

```
//Gives individual access to a specific system constant
private static void AccessSystemConstant(PlcTagTable tagTable)
{
    PlcTag systemConstant = tagTable.SystemConstants.Find("Constant XYZ");
    // The parameter specifies the name of the tag
}
```

Siehe auch

- Benutzerdefinierte Gruppen für PLC-Variablen erzeugen (Seite 603)
- Benutzerdefinierte Gruppen für PLC-Variablen löschen (Seite 604)
- PLC-Variablen-tabelle aus einer Gruppe löschen (Seite 608)
- Auf PLC-Variablen zugreifen (Seite 609)

Starten des PLC-Variableneditors (Seite 600)

Zeitpunkt der letzten Änderung einer PLC-Variablentabelle lesen (Seite 607)

5.11.6 Funktionen für Software-Units

5.11.6.1 Auf Software-Units zugreifen

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness auf die Units zugreifen. Die Units sind wichtige Bestandteile der PLC-Programmierung. Units werden nur von bestimmten PLCs unterstützt. Deshalb werden die Units im Objektmodell nicht statisch modelliert, sodass sie im PLC-Softwarecontainer nicht direkt auffindbar sind.

Der PlcUnitProvider schafft die Zugriffsmöglichkeit auf Units mit Hilfe von `GetService()`; hierüber kann auch auf den Provider selbst zugegriffen werden. Der PlcUnitProvider ermöglicht den Zugriff auf die PlcUnitSystemGroup über den UnitGroup navigator, der die PlcUnitComposition mit den spezifischen Möglichkeiten der allgemeinen Zusammensetzung enthält (Iterieren, Finden, Erstellen, Importieren usw. in Bezug auf PlcUnits).

Die Darstellung der Unit (PlcUnit) enthält die eigenen spezifischen Eigenschaften der Unit als Openness-Attribute.

Sie können auf das folgende Attribut und die Methode für Eigenschaften von Openness-Units zugreifen:

Attributname	Typ
Author	String
Name	String

Methodenname	Rückgabetyt
Export	void
Delete	void
GetAttributes	IList<object>
SetAttributes	void

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Unter der PlcUnit können Sie nacheinander die folgenden Objekte durchlaufen:

- Kommentar: MultilingualText
- die BlockGroup (PlcBlockSystemGroup)
 - unit.BlockGroup
- TagTableGroup (PlcTagTableSystemGroup)
 - unit.TagTableGroup
- TypeGroup (PlcTypeSystemGroup)
 - unit.TypeGroup

Programmcode

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

```

private static PlcUnit AccessSoftwareUnit(Project project, string plcName = "PLC_1", string
plcUnitName = "Unit_1")
{
    PlcUnit plcUnit = null;
    Device device = null;
    DeviceItem deviceItem = null;
    if (project == null)
    {
        Console.WriteLine("Project cannot be null");
        return plcUnit;
    }
    //Finding the plc
    Console.WriteLine(String.Format("Iterate through {0} device(s)", project.Devices.Count));
    foreach (Device checkDevice in project.Devices)
    {
        if (device == null)
        {
            Console.WriteLine(String.Format("Device: \"{0}\".", checkDevice.Name));
            foreach (DeviceItem checkDeviceItem in checkDevice.DeviceItems)
            {
                if (device == null)
                {
                    if (checkDeviceItem.Name == plcName)
                    {
                        device = checkDevice;
                        deviceItem = checkDeviceItem;
                    }
                }
            }
        }
        if (device == null)
        {
            Console.WriteLine("Device cannot be null");
            return plcUnit;
        }
        else
        {
            SoftwareContainer softwareContainer =
                ((IEngineeringServiceProvider)deviceItem).GetService<SoftwareContainer>();
            PlcSoftware software = softwareContainer.Software as PlcSoftware;
            //Find the SoftwareUnit
            PlcUnitProvider plcUnitProvider = software.GetService<PlcUnitProvider>();
            plcUnit = plcUnitProvider.UnitGroup.Units.Find(plcUnitName);
            if (plcUnit != null)
            {
                //Do something
                UnitAccessType att = readBlockAttribute(plcUnit);
                IList<object> atts = readBlockAttributeList(plcUnit);
            }
        }
        return plcUnit;
    }
}

```

Siehe auch

Projekt öffnen (Seite 131)

5.11.6.2 Mit Software-Units arbeiten**Voraussetzung**

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Die Units sind wichtige Bestandteile der PLC-Programmierung und in der PlcSoftware zu finden.

Der unit provider wird über GetService aus der PlcSoftware abgerufen, und daraus kann wiederum die PlcUnitComposition abgerufen werden.

Mit Units können Sie die folgenden grundlegenden Aktionen durchführen, während Sie TIA Portal verwenden:

- Units erstellen
- Units löschen
- Units umbenennen

Programmcode: Units erstellen

Ändern Sie folgenden Programmcode, um Units zu erstellen.

```
private static void CreateUnit(PlcUnitProvider provider, string plcUnitName)
{
    PlcUnitComposition unitComposition = provider.UnitGroup.Units;
    //Creating Unit
    PlcUnit unit1 = unitComposition.Create(plcUnitName);
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Wenn Sie versuchen, eine Unit mit einem Namen zu erstellen, der bereits verwendet wird, wird eine Fehlermeldung ausgegeben. Eine Recoverable Exception wird mit folgender Meldung ausgelöst: "Unit mit dem Namen 'Unit1' ist bereits vorhanden."

```
PlcUnit unit1 = unitComposition.Create("Unit1");  
...  
...  
try  
{  
    PlcUnit unit2 = unitComposition.Create("Unit1");  
}  
catch (EngineeringTargetInvocationException e)  
{  
    Console.WriteLine(e.Message);  
}
```

Ausnahmebehandlung: Units erstellen

Wenn Sie den Namen falsch eingeben, z.B. mit einem Leerzeichen am Anfang, ungültigen Zeichen oder bei zu langem Namen, wird eine Fehlermeldung ausgegeben: Eine Recoverable Exception wird mit folgender Fehlermeldung ausgelöst: "Der Wert des Attributs 'Name' enthält ein ungültiges Zeichen an Position 0."

```
private static void CreateUnit(PlcUnitProvider provider, string plcUnitName)  
{  
    PlcUnitComposition unitComposition = provider.UnitGroup.Units;  
    //Creating Unit  
    try  
    {  
        PlcUnit unit1 = unitComposition.Create(plcUnitName);  
    }  
    catch (EngineeringTargetInvocationException e)  
    {  
        Console.WriteLine(e.Message);  
    }  
}
```

Wenn Sie versuchen, eine Unit mit einem Namen zu erstellen, der bereits verwendet wird, wird eine Fehlermeldung ausgegeben. Eine Recoverable Exception wird mit folgender Meldung ausgelöst: "Unit mit dem Namen 'Unit1' ist bereits vorhanden."

```
private static void CreateUnit(PlcUnitProvider provider, string plcUnitName)
{
    PlcUnitComposition unitComposition = provider.UnitGroup.Units;
    try
    {
        PlcUnit unit2 = unitComposition.Create("Unit1");
    }
    catch (EngineeringTargetInvocationException e)
    {
        Console.WriteLine(e.Message);
    }
}
```

Programmcode: Units löschen

Ändern Sie folgenden Programmcode, um Units zu löschen:

```
private static void DeleteUnit(PlcUnitProvider unitProvider, string plcUnitName)
{
    unitProvider.UnitGroup.Units.Find(plcUnitName).Delete();
}
```

Es wird eine `NullReferenceException` ausgegeben, wenn die Methode `Find()` aus dem obigen Beispiel keinen Verweis auf eine Unit bereitstellt.

Programmcode: Units umbenennen

Sie können die Attribute auf mehrere Arten festlegen. Die Festlegung kann durch Zuordnung eines Werts oder durch Aufruf von `SetAttribute` oder `SetAttributes` erfolgen. Die Eigenschaft `Name` kann durch `GetAttribute` oder `GetAttributes` validiert werden. Damit Sie `SetAttribute` oder `GetAttribute` verwenden können, muss das Objekt vorher an `IEngineeringObject` übertragen werden.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie folgenden Programmcode, um Units umzubenennen.

```
private void RenameUnit(PlcUnitProvider provider)
{
    PlcUnitComposition unitComposition = provider.UnitGroup.Units;
    //Set Value
    PlcUnit unit1 = unitComposition.Create("Unit1");
    unit1.Name = "Unit1_new";
    //Using SetAttributes():
    var attrList = new List<KeyValuePair<string, object>>()
    {
        new KeyValuePair<string, object>("Name", "Unit2_new")
    };
    unit1.SetAttributes(attrList);
    //Using SetAttribute()
    PlcUnit unit2 = unitComposition.Create("Unit2");
    IEngineeringObject unit = (IEngineeringObject)unit2;
    unit.SetAttribute("Name", "Unit2_new");
}
```

Wenn Sie den Namen falsch eingeben, z. B. mit einem Leerzeichen am Anfang oder mit ungültigen Zeichen, wird eine Fehlermeldung ausgegeben. Eine Recoverable Exception wird mit folgender Fehlermeldung ausgelöst: "Der Wert von Attribut 'Name' enthält ein ungültiges Zeichen an Position 0."

```
PlcUnit unit1 = unitComposition.Create("Unit_1");
try
{
    unit1.Name = "Unit_1";
}
catch (EngineeringTargetInvocationException e)
{
    Console.WriteLine(e.Message);
}
```

Wenn Sie versuchen, eine Unit in einen Namen umzubenennen, der bereits verwendet wird, wird eine Fehlermeldung ausgegeben. Eine wiederherstellbare Ausnahme wird mit folgender Fehlermeldung ausgelöst: "Die Eigenschaft 'Name' hat einen ungültigen Wert. 'Unit1'. Eine Software-Unit mit demselben Namen ist bereits vorhanden."

```
PlcUnit unit1 = unitComposition.Create("Unit1");
PlcUnit unit2 = unitComposition.Create("Unit_2");
try
{
    PlcUnit unit2 = unitComposition.Create("Unit1");
}
catch (EngineeringTargetInvocationException e)
{
    Console.WriteLine(e.Message);
}
```


Siehe auch

Projekt öffnen (Seite 131)

5.11.6.3 Auf Software-Units zugrundeliegende Objekte zugreifen**Voraussetzung**

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness zugreifen:

- Rekursiv auf die Systemgruppe "Programmbausteine", die darin enthaltenen Gruppen und Bausteine
- Rekursiv auf die Systemgruppe "PLC-Datentypen", die darin enthaltenen Gruppen und Datentypen
- Rekursiv auf die Systemgruppe "PLC-Variablen", die darin enthaltenen Gruppen und Variablen

Programmcode: Auf Programmbausteine und Bausteingruppen zugreifen

Um auf den Programmbaustein unter der aktuellen PLC-Unit zuzugreifen, ändern Sie folgenden Programmcode, indem Sie die Zusammenstellung der Programmbausteine abrufen und die darin enthaltenen Bausteine durchiterieren:

```
private void AccessProgramBlock(PlcUnit m_PlcUnit)
{
    PlcBlockComposition blockComposition = m_PlcUnit.BlockGroup.Blocks;
    foreach (PlcBlock block in blockComposition)
    {
        //usage of block
    }
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Um auf die Programmbausteingruppe unter der aktuellen PLC-Unit zuzugreifen, ändern Sie folgenden Programmcode, indem Sie die Zusammenstellung der Programmbausteingruppen abrufen und die darin enthaltenen Gruppen durchiterieren:

```
private void AccessProgramBlockGroup(PlcUnit m_plcUnit)
{
    PlcBlockUserGroupComposition usergroupComposition = m_PlcUnit.BlockGroup.Groups;
    foreach (PlcBlockUserGroup group in usergroupComposition)
    {
        PlcBlockComposition blockComposition = group.Blocks;
        foreach (PlcBlock block in blockComposition)
        {
            //...
            //usage of the block
            //...
        }
    }
}
```

Programmcode: Auf PLC-Datentypen und Datentypgruppen zugreifen

Um auf die PLC-Datentypen unter der aktuellen PLC-Unit zuzugreifen, ändern Sie folgenden Programmcode, indem Sie die Zusammenstellung der PLC-Datentypen abrufen und die darin enthaltenen Datentypen durchiterieren:

```
private void AccessPlcDatatypes(PlcUnit m_PlcUnit)
{

    PlcTypeComposition typeComposition = m_PlcUnit.TypeGroup.Types;
    foreach (PlcType type in typeComposition)
    {
        // ...
        // usage of the type
        // ...
    }
}
```

Um auf die PLC-Datentypgruppe unter der aktuellen PLC-Unit zuzugreifen, ändern Sie folgenden Programmcode, indem Sie die Zusammenstellung der PLC-Datentypen abrufen und die darin enthaltenen Gruppen durchiterieren:

```
private void AccessPlcDatatypeGroup(PlcUnit m_PlcUnit)
{
    PlcTypeUserGroupComposition usergroupComposition = m_PlcUnit.TypeGroup.Groups;
    foreach (PlcTypeUserGroup group in usergroupComposition)
    {
        PlcTypeComposition typeComposition = group.Types;
        foreach (PlcType type in typeComposition)
        {
            // ...
            // usage of the type
            // ...
        }
    }
}
```

Programmcode: Auf PLC-Variablentabellen und Variablentabellengruppen zugreifen

Um auf die PLC-Variablentabellen unter der aktuellen PLC-Unit zuzugreifen, ändern Sie folgenden Programmcode, indem Sie die Zusammenstellung der PLC-Variablentabellen abrufen und die darin enthaltenen Variablentabellen durchiterieren:

```
private void AccessPlcTagTables(PlcUnit m_PlcUnit)
{
    PlcTagTableComposition tagtableComposition = m_PlcUnit.TagTableGroup.TagTables;
    foreach (PlcTagTable type in tagtableComposition)
    {
        //...
        // usage of the tag table
        //...
    }
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Um auf die PLC-Variablentabellengruppen unter der aktuellen PLC-Unit zuzugreifen, ändern Sie folgenden Programmcode, indem Sie die Zusammenstellung der PLC-Variablentabellengruppen abrufen und die darin enthaltenen Gruppen durchiterieren:

```
private void AccessPlcTagTableGroups(PlcUnit m_PlcUnit)
{
    PlcTagTableUserGroupComposition usergroupComposition = m_PlcUnit.TagTableGroup.Groups;
    foreach (PlcTagTableUserGroup group in usergroupComposition)
    {
        PlcTagTableComposition tagtableComposition = group.TagTables;
        foreach (PlcTagTable type in tagtableComposition)
        {
            //...
            // usage of the tag table
            //...
        }
    }
}
```

5.11.6.4 Auf vorhandene Beziehungen einer Unit zugreifen

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness auf alle vorhandenen Beziehungen einer Unit zugreifen und so die Eigenschaften dieser Beziehungen lesen.

Die folgenden Attribute werden von der Unitbeziehung in Openness unterstützt:

Attributname	Typ	Beschreibung
RelationType	UnitRelationType	Art der Beziehung
RelatedObject	String	Enthält den Namen des zugänglichen Elements

Die folgenden ENUM-Werte werden für das Attribut RelationType bereitgestellt:

- Software Unit
- Non-Unit DB
- TO DB

Programmcode: Auf Beziehungen zugreifen

Beziehungen, die Beziehungsarten und die Namen der bezogenen Objekte können auf mehrere Arten abgerufen werden.

Ändern Sie folgenden Programmcode, um die Zusammensetzung einer Beziehung abzurufen:

```
private void AccessRelationComposition(PlcUnitProvider provider)
{
    //assuming existing units
    m_PlcUnit = provider.UnitGroup.Units[0];
    PlcUnitRelationComposition unitRelations = m_PlcUnit.Relations;
}
```

Ändern Sie folgenden Programmcode, um Beziehungen nach Indexer abzurufen:

```
private void AccessRelationByIndex(PlcUnitProvider provider)
{
    m_PlcUnit = provider.UnitGroup.Units[0]; //assuming existing units
    PlcUnitRelation unitRelation = m_PlcUnit.Relations[1];
}
```

Ändern Sie folgenden Programmcode, um Beziehungen durch Iteration abzurufen:

```
private void AccessUnitRelations(PlcUnitProvider provider)
{
    //assuming existing units
    m_PlcUnit = provider.UnitGroup.Units[0];
    PlcUnitRelationComposition unitRelations = m_PlcUnit.Relations;
    foreach (PlcUnitRelation relation in unitRelations)
    {
        // using 'relation'
    }
}
```

Ändern Sie folgenden Programmcode, um den Beziehungstyp einer Unit durch Attribut RelationType abzurufen:

```
private void AccessUnitRelationType(PlcUnitProvider provider)
{
    //assuming existing units
    m_PlcUnit = provider.UnitGroup.Units[0];
    UnitRelationType unitRelationType = m_PlcUnit.Relations[2].RelationType;
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie folgenden Programmcode, um den Namen des bezogenen Objekts einer Unit durch das Attribut `RelatedObject` abzurufen:

```
private void AccessUnitRelatedObject(PlcUnitProvider provider)
{
    //assuming existing units
    m_PlcUnit = provider.UnitGroup.Units[0];
    string unitRelatedObjectName = m_PlcUnit.Relations[1].RelatedObject;
}
```

Ändern Sie folgenden Programmcode, um eine Beziehung mit Hilfe von "Suchen" mit dem Namen des zugänglichen Elements (`RelatedObject`) zu finden:

```
private void SearchUnitRelationFromRelatedObject(PlcUnitProvider provider)
{
    //assuming existing units
    m_PlcUnit = provider.UnitGroup.Units[0];
    PlcUnitRelation relation = m_PlcUnit.Relations.Find(unitRelatedObjectName);
}
```

5.11.6.5 Eigenschaften von Software-Units aktualisieren

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Units-Eigenschaften wie z. B. Autor und Kommentar aktualisieren. Die Eigenschaft 'Kommentar' kann nicht direkt festgelegt und abgerufen werden, weil es sich dabei um mehrsprachigen Text handelt. Die Eigenschaft "Text" des Kommentars kann aber aktualisiert werden.

Das `Comment.Item` enthält die verschiedenen Kulturen ("cultures") des TIA Portal-Projekts. Es kann über die Zuordnung von Werten festgelegt werden. Die Projektsprachen sind in der `Comment.Item` composition indexiert, wobei mit 0 die erste Projekt-Standardsprache angegeben wird. Wenn mehrere Sprachen anzugeben sind, können Sie die Liste durchlaufen. Ansonsten wird, wenn die Kultur nicht vorhanden ist, eine `EngineeringTargetException` ausgelöst.

Programmcode: Eigenschaft "Author" der Unit aktualisieren

Ändern Sie folgenden Programmcode, um die Eigenschaft "Author" der Unit zu aktualisieren.

```
private static void UpdateUnitProperty(PlcUnit unit1, PlcUnit unit2, string newAuthor =
"Z012345")
{
    //Set value
    unit1.Author = newAuthor;
    //Using SetAttributes()
    var attrList = new List<KeyValuePair<string, object>>()
    {
        new KeyValuePair<string, object>("Author", newAuthor)
    };
    unit1.SetAttributes(attrList);
    //Using SetAttribute()
    IEngineeringObject unit = (IEngineeringObject)unit2;
    unit.SetAttribute("Author", newAuthor);
}
```

Fehlermeldung

Wenn versucht wird, einen falschen Wert einzugeben, z. B. mit Leerzeichen am Anfang, oder wenn der Wert unzulässige Zeichen enthält, wird eine Fehlermeldung ausgegeben. Eine Recoverable Exception wird mit folgender Fehlermeldung ausgelöst: "Der Wert von Attribut 'Author' enthält ein ungültiges Zeichen an Position 0."

```
private static void UpdateUnitProperty(PlcUnitProvider provider, string plcUnitName)
{
    PlcUnitComposition unitComposition = provider.UnitGroup.Units;
    //Creating Unit
    PlcUnit unit1 = unitComposition.Create(plcUnitName);
    try
    {
        unit1.Author = " Author";
    }
    catch (EngineeringTargetInvocationException e)
    {
        Console.WriteLine(e.Message);
    }
}
```

Programmcode: Eigenschaft "Comment" der Unit aktualisieren

Die Eigenschaft 'Comment' der Unit kann nicht direkt festgelegt oder abgerufen werden, weil es sich dabei um MultilingualText handelt. Sie können aber die Eigenschaft "Text" des Kommentars aktualisieren. Das Comment.Item enthält die verschiedenen Kulturen ("cultures") des TIA Projekts. Es kann über die Zuordnung von Werten festgelegt werden. Die Projektsprachen sind in der Comment.Item composition indexiert, wobei mit 0 die erste Projekt-Standardsprache angegeben wird. Wenn mehrere Sprachen angegeben sind, können Sie die Liste durchlaufen. Ansonsten wird, wenn die Kultur nicht vorhanden ist, eine EngineeringTargetException ausgelöst.

```
private static void UpdateUnitProperty(PlcUnit unit1)
{
    //Setting the default first culture comment Item text:
    unit1.Comment.Items[0].Text = "new Comment";
    //Setting other culture comment Item Text:
    unit1.Comment.Items[1].Text = "neuro Kommentar";
}
```

Fehlermeldung

Wenn versucht wird, SetAttribute, SetAttributes, GetAttribute, GetAttributes. zu verwenden, wird eine EngineeringNotSupportedException mit folgender Fehlermeldung ausgegeben: "Kommentar wird durch Typ 'Siemens.Engineering.SW.Units.PlcUnit' nicht unterstützt."

```
private static void UpdateUnitProperty(PlcUnit unit1)
{
    try
    {
        ((IEngineeringObject)unit1).SetAttribute("Comment", "new comments");
    }
    catch (EngineeringNotSupportedException e)
    {
        Console.WriteLine(e.Message);
    }
}
```

Wenn versucht wird, eine Beziehung zu einer nicht vorhandenen Kultur herzustellen, wird eine Fehlermeldung ausgegeben. Eine EngineeringTargetException wird mit

folgender Fehlermeldung ausgelöst: "Das Argument 'index' (3) liegt außerhalb des gültigen Wertebereichs."

```
private static void UpdateUnitProperty(PlcUnit unit1)
{
    try
    {
        unit1.Comment.Items[3].Text = "new Comment";
    }
    catch (EngineeringTargetInvocationException e)
    {
        Console.WriteLine(e.Message)
    }
}
```

5.11.6.6 Objekt einer Software-Unit veröffentlichen

Voraussetzung

- Die Anwendung ist über TIA Portal Openness mit TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness das Attribut "Access" zugrunde liegender Objekte von Units abrufen und einstellen, so dass Sie deren Zugänglichkeit zwischen Units beeinflussen können.

In Openness-Units werden die folgenden Objekte unterstützt:

- Programmbausteine und Datenbausteine – außer OBs
- PLC-Typen

Der folgende mögliche Wert des Attributs Access dargestellt als Enumeration UnitAccessType:

- Veröffentlicht
- Unveröffentlicht

Hinweis

Verfügbar ist das Attribut "Access" nur an Objekten unter einer Unit. Wenn es also nicht unter mit zu PlcUnit gehörenden Objekten abgerufen oder eingestellt werden soll, wird eine EnginneringNotSupportedException ausgelöst. Im Fall eines falschen Werts (z. B. ungültige Enumerationswerte) wird eine EngineeringTargetInvocationException ausgelöst.

Programmcode: Zugriffsattribut von PLC-Bausteinen konfigurieren

```
private void AccessAttributeOfPLCBlocks(PlcUnit plcUnit, string blockName = "FB_1")
{
    //Getting attribute value with GetAttribute
    ...
    PlcBlock block = plcUnit.BlockGroup.Find(blockName);
    UnitAccessType currentAccessValue = (UnitAccessType)block.GetAttribute("Access");
    ...
    //Getting attribute value with GetAttributes
    ...
    PlcBlock block = plcUnit.BlockGroup.Blocks.Find(blockName);
    List<string> attrList = new List<string>();
    {
        "Access"
    };
    IList<object> getAttributeValues = block.GetAttributes(attrList));
    ...
    //Setting attribute value with SetAttribute
    ...
    PlcBlock block = plcUnit.BlockGroup.Blocks.Find(blockName);
    UnitAccessType newAccessValue = UnitAccessType.Published;
    block.SetAttribute("Access", newAccessValue);
    ...
    //Setting attribute value with SetAttributes
    ...
    PlcBlock block = plcUnit.BlockGroup.Blocks.Find(blockName);
    UnitAccessType newAccessValue = UnitAccessType.Published;
    IList attrList = new List<KeyValuePair<string, object>>()
    {
        new KeyValuePair<string, object>("Access", newAccessValue),
    };
    block.SetAttributes(attrList);
    ...
}
```

Programmcode: Zugriffsattribut von PLC-Bausteinen konfigurieren

```
private void AccessAttributeOfPLCTypes(PlcUnit plcUnit, string typeName = "UDT_1")
{
    //Getting attribute value with GetAttribute
    PlcType type = PlcUnit.TypeGroup.Types.Find(typeName);
    UnitAccessType currentAccessValue = (UnitAccessType)type.GetAttribute("Access");
    //Getting attribute value with GetAttributes
    PlcType type = m_PlcUnit.TypeGroup.Types.Find(typeName);
    List<string> attrList = new List<string>()
    {
        "Access"
    };
    IList<object> getAttributeValues = type.GetAttributes(attrList));
    //Setting attribute value with SetAttribute
    PlcType type = m_PlcUnit.TypeGroup.Types.Find(typeName);
    UnitAccessType newAccessValue = UnitAccessType.Published;
    type.SetAttribute("Access", newAccessValue);
    //Setting attribute value with SetAttributes
    PlcType type = m_PlcUnit.TypeGroup.Types.Find(typeName);
    UnitAccessType newAccessValue = UnitAccessType.Published;
    IList attrList = new List<KeyValuePair<string, object>>()
    {
        new KeyValuePair<string, object>("Access", newAccessValue),
    };
    type.SetAttributes(attrList));
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
 Projekt öffnen (Seite 131)

5.11.6.7 Externe Quellen in Units hinzufügen**Voraussetzung**

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
 Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
 Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness eine externe Quelle unter einer Software-Unit hinzufügen.
 Sie können die Quelldateien wie .SCL, .DB, .UDT unter Units hinzufügen.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Mit der Openness API können Sie die folgenden Aufgaben mit ExternalSourceGroup unter Software-Units ausführen:

- ExternalSourceGroup unter Units hinzufügen
- Bausteine/UDTs aus einer externen Quelle unter Software-Units generieren
- Quelle aus Bausteinen/UDTs unter Units exportieren

Programmcode: ExternalSourceGroup hinzufügen

Ändern Sie folgenden Programmcode, um eine externe Quelle unter einer Software-Unit hinzuzufügen:

```
private void AddExternalSourceGroup(PlcUnitProvider unitProvider, string plcUnitName =
"Unit1")
{
    PlcUnit newUnit = unitProvider.UnitGroup.Units.Create(plcUnitName);
    PlcExternalSource unitExternalSource =
    newUnit.ExternalSourceGroup.ExternalSources.CreateFromFile(externalSourceFilename,
    GetFilePath(externalSourceFilename));
}
```

Programmcode: Bausteine/UDTs aus Quelle generieren

Ändern Sie folgenden Programmcode, um Bausteine/UDTs mit leeren Rückgabetypen aus einer externen Quelle unter einer Software-Unit zu generieren:

```
private void GenerateBlocksFromSource(PlcExternalSource unitExternalSource)
{
    unitExternalSource.GenerateBlocksFromSource();
}
```

Ändern Sie folgenden Programmcode, um Bausteine/UDTs mit einer aus einer externen Quelle abgerufenen Bausteinsammlung unter einer Software-Unit zu generieren:

```
private static IList<IEngineeringObject> GenerateBlocksFromSource(PlcExternalSource
unitExternalSource)
{
    // Blocks/UDTs gets generated only if there are no compilation errors
    IList<IEngineeringObject> generatedBlocksUnderUnitList =
    unitExternalSource.GenerateBlocksFromSource(GenerateBlockOption.None);
    // Blocks/UDTs will be generated regardless of any compilation errors
    IList<IEngineeringObject> generatedBlocksUnderUnitList =
    unitExternalSource.GenerateBlocksFromSource(GenerateBlockOption.KeepOnError);
}
```

Programmcode: Quelle aus Bausteinen/UDTs exportieren

Ändern Sie folgenden Programmcode, um eine Quelle aus Bausteinen unter einer Software-Unit zu generieren:

```
private void GenerateSourceFromBlocks(PlcUnit newUnit)
{
    // For blocks
    PlcBlock unitPlcBlock = newUnit.BlockGroup.Blocks.Find(generatedBlockName);
    // Generate source from blocks with dependencies
    newUnit.ExternalSourceGroup.GenerateSource(new[]{unitPlcBlock}, new
    FileInfo(outputFileGeneratedPath), GenerateOptions.WithDependencies);
    // Generate source from blocks without dependencies
    newUnit.ExternalSourceGroup.GenerateSource(new[]{unitPlcBlock}, new
    FileInfo(outputFileGeneratedPath), GenerateOptions.None);
}
```

Ändern Sie folgenden Programmcode, um eine Quelle aus UDTs unter einer Software-Unit zu generieren:

```
private void GenerateSourceFromUDTs(PlcUnit newUnit)
{
    // For UDTs
    PlcType unitPlcUdt = newUnit.TypeGroup.Types.Find(generatedUdtName);
    //Generate source from UDTs with dependencies
    newUnit.ExternalSourceGroup.GenerateSource(new[]{unitPlcUdt}, new
    FileInfo(outputFileGeneratedPath), GenerateOptions.WithDependencies);
    //Generate source from UDTs without dependencies
    newUnit.ExternalSourceGroup.GenerateSource(new[]{unitPlcUdt}, new
    FileInfo(outputFileGeneratedPath), GenerateOptions.None);
}
```

Programmcode: Externe Quelldateigruppe enumerieren

Ändern Sie folgenden Programmcode, um die Zusammenstellung der externen Quelldateigruppe unter der aktuellen Unit zu enumerieren:

```
private void AccessExternalSourceFileGroup(PlcUnitProvider unitProvider)
{
    PlcUnit newUnit = unitProvider.UnitGroup.Units.Find(textBoxAddNewUnit.Text);
    PlcExternalSourceComposition unitExtSrcComposition =
    newUnit.ExternalSourceGroup.ExternalSources;
    foreach (PlcExternalSource unitExtSrc in unitExtSrcComposition)
    {
        unitExtSrc.GenerateBlocksFromSource(GenerateBlockOption.None);
    }
}
```

5.11.6.8 Units als Masterkopien

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)
- Eine Unit ist erstellt
Siehe Mit Software-Units arbeiten (Seite 617)

Einleitung

Sie können mit TIA Portal Openness Units als Masterkopien in die Projektbibliothek und in die globale Bibliothek kopieren.

Mit Units als Masterkopie können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Eine Unit als Masterkopie in der Projektbibliothek aus Software-Units erstellen
- Eine Unit als Masterkopie in der globalen Bibliothek aus Software-Units erstellen
- Eine Unit aus einer Masterkopie in der Projektbibliothek in Software-Units neu erstellen
- Eine Unit aus einer Masterkopie in der globalen Bibliothek in Software-Units neu erstellen

Programmcode

Ändern Sie folgenden Programmcode, um eine neue Unit als Masterkopie in der Projektbibliothek aus Software-Units mit Hilfe von `MasterCopyComposition` zu erstellen.

```
private void IMasterCopySource CreateUnitAsMastercopyInProjectLibrary(PlcUnitProvider
m_UnitProvider, string plcUnitName = "Unit_1")
{
    PlcUnitProvider m_UnitProvider = plc.GetService<PlcUnitProvider>();
    PlcUnitComposition m_UnitComposition = m_UnitProvider.UnitGroup.Units;
    PlcUnit m_SoftwareUnit1 = unitComposition.Create("Unit_1");//Assuming existing units
    IMasterCopySource m_UnitAsMasterCopy = (IMasterCopySource)m_SoftwareUnit1;//Assuming that
    m_SoftwareUnit1 is present in the PLC
    m_ProjectLibrary.MasterCopyFolder.MasterCopies.Create(m_UnitAsMasterCopy);
}
```

Ändern Sie folgenden Programmcode, um eine Unit als Masterkopie in der globalen Bibliothek aus Software-Units zu erstellen.

```
private void IMasterCopySource CreateUnitAsMasterCopyInGlobalLibrary(PlcUnit
m_SoftwareUnit1, TiaPortal m_TiaPortal)
{
// ...
IMasterCopySource m_UnitAsMasterCopy = (IMasterCopySource)m_SoftwareUnit1;//Assuming that
m_SoftwareUnit1 is present in the PLC
m_TiaPortal.GlobalLibraries.Open(libfile, OpenMode.ReadWrite);
GlobalLibrary m_GlobalLibrary = m_TiaPortal.GlobalLibraries[0];
m_GlobalLibrary.MasterCopyFolder.MasterCopies.Create(m_UnitAsMasterCopy)
// ...
}
```

Wenn versucht wird, eine neue Unit in der schreibgeschützten globalen Bibliothek zu erstellen, wird eine Fehlermeldung ausgegeben. Eine wiederherstellbare Ausnahme wird mit der Meldung "Schreiben in schreibgeschützte Bibliotheken nicht möglich" ausgelöst.

```
private void IMasterCopySource CreateUnitAsMasterCopyInGlobalLibrary(PlcUnit
m_SoftwareUnit1, TiaPortal m_TiaPortal)
{
IMasterCopySource m_UnitAsMasterCopy = (IMasterCopySource)m_SoftwareUnit1;//Assuming that
m_SoftwareUnit1 is present in the PLC
m_TiaPortal.GlobalLibraries.Open(libfile, OpenMode.ReadOnly);
GlobalLibrary m_GlobalLibrary = m_TiaPortal.GlobalLibraries[0];
try
{
m_GlobalLibrary.MasterCopyFolder.MasterCopies.Create(m_UnitAsMasterCopy);
}
catch (Exception e)
{
Console.WriteLine(e.Message);
}
}
```

Ändern Sie folgenden Programmcode, um eine Unit aus einer Masterkopie in der Projektbibliothek in Software-Units in PNV neu zu erstellen:

```
private void RecreateUnitFromProjectLibrary(Project project, PlcSoftware software, string
plcUnitName = "Unit_2")
{
//...
ProjectLibrary m_ProjectLibrary = project.ProjectLibrary;
PlcUnitProvider m_UnitProvider = software.GetService<PlcUnitProvider>();
PlcUnitComposition m_UnitComposition = m_UnitProvider.UnitGroup.Units;
// ...
MasterCopy mc_Unit_2 = m_ProjectLibrary.MasterCopyFolder.MasterCopies.Find(plcUnitName);
m_UnitComposition.CreateFrom(mc_Unit_2);//Recreate a Unit from Project Library to
SoftwareUnits folder
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie folgenden Programmcode, um eine Unit aus einer Masterkopie in der globalen Bibliothek in Software-Units in PNV neu zu erstellen:

```
private void RecreateUnitFromGlobalLibrary(TiaPorta m_TiaPortal, string plcUnitName =
"Unit_2")
{
// ...
GlobalLibrary m_GlobalLibrary = m_TiaPortal.GlobalLibraries[0];
// ...
mc_Unit_2 = m_GlobalLibrary.MasterCopyFolder.MasterCopies.Find("Unit_2");
m_UnitComposition.CreateFrom(mc_Unit_2); //Recreate a unit from Global Library to
SoftwareUnits folder
}
```

5.11.6.9 Vorhandene Beziehungen aktualisieren und Beziehungen erstellen/löschen

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Beziehungen von PLC-Units ändern und so die Zugänglichkeit der Objekte in anderen PLC-Units oder außerhalb von PLC-Units steuern.

Mit TIA Portal Openness können Sie die folgenden Arten von Änderungen vornehmen:

- Neue Beziehungen erstellen
- Vorhandene Beziehungen löschen
- Vorhandene Beziehungen ändern

Programmcode: Neue Beziehungen erstellen

Sie können eine neue Beziehung erstellen, indem Sie die entsprechende Beziehungsart und den Namen des bezogenen Objekts angeben.

```
private void CreateNewRelation(PlcSoftware plcTarget)
{
private PlcUnitRelation m_SoftwareUnitRelation;
private PlcUnitRelation m_NonUnitDBRelation;
private PlcUnitRelation m_TODDBRelation;
//...
PlcUnitProvider plcUnitProvider = plcTarget.GetService<PlcUnitProvider>();
m_PlcUnit = plcUnitProvider.UnitGroup.Units[0]; //assuming existing units
}
```


Ändern Sie folgenden Programmcode, um eine neue Beziehung zu erstellen, mit der Sie auf ein Objekt in einer anderen PLC-Unit zugreifen:

```
private void CreateRelationForAccessObject(PlcUnit m_PlcUnit, string plcUnitName =
"Unit_2")
{
// ...
//assuming Plc unit "Unit_2" is already existing in the Plc
m_SoftwareUnitRelation = m_PlcUnit.Relations.Create(plcUnitName,
UnitRelationType.SoftwareUnit);
// ...
}
```

Ändern Sie folgenden Programmcode, um eine neue Beziehung zu erstellen, mit der Sie auf einen PLC-Datenbaustein außerhalb von PLC-Units zugreifen:

```
private void CreateRelationForAccessDataBlock(PlcUnit m_PlcUnit, string dataBlockName =
"Data_block_1")
{
// ...
//assuming Plc data block "Data_block_1" is already existing in the Plc
m_NonUnitDBRelation = m_PlcUnit.Relations.Create(dataBlockName,
UnitRelationType.NonUnitDB);
// ...
}
```

Ändern Sie folgenden Programmcode, um eine neue Beziehung zu erstellen, mit der Sie auf ein Technologieobjekt außerhalb von PLC-Units zugreifen:

```
private void CreateRelationForAccessTechnologicalObject(PlcUnit m_PlcUnit, string
technologicalObjectName = "SpeedAxis_1")
{
// ...
//assuming Technological object "SpeedAxis_1" is already existing in the Plc
m_TODBRelation = m_PlcUnit.Relations.Create(technologicalObjectName,
UnitRelationType.TODB);
// ...
}
```

Mögliche Fehlerszenarios beim Erstellen einer neuen Beziehung:

Ändern Sie folgenden Programmcode, um eine neue Beziehung zu erstellen, mit der auf ein nicht vorhandenes bezogenes Objekt zugegriffen wird:

```
private void CreateRelationForAccessNonExistingRelatedObject(PlcUnit m_PlcUnit)
{
//...
//assuming Plc unit "NonExistingUnit" is not existing in the Plc yet
m_SoftwareUnitRelation = m_PlcUnit.Relations.Create("NonExistingUnit", UnitRelationType.
SoftwareUnit);
//...
}
```

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Hinweis

Im obigen Programmcode wird die Beziehung erstellt und keine Ausnahme ausgelöst.

Ändern Sie folgenden Programmcode, um eine neue Beziehung zu erstellen, indem Sie einen Namen für das bezogene Objekt angeben, der nicht den TIA-Benennungsregeln entspricht:

```
private void CreateRelationForRelatedObject(PlcUnit m_PlcUnit)
{
    //...
    m_SoftwareUnitRelation = m_PlcUnit.Relations.Create(" Unit_2",
    UnitRelationType.SoftwareUnit);
    //...
}
```

Hinweis

Im obigen Programmcode wird die Beziehung nicht erstellt und es wird eine wiederherstellbare Ausnahme aufgrund von führenden Leerzeichen ausgelöst.

Ändern Sie folgenden Programmcode, um eine neue Beziehung von der Unit mit sich selbst zu erstellen:

```
private void CreateRelationFromUnit(PlcUnit m_PlcUnit)
{
    //...
    //assuming m_PlcUnit is assigned to Plc unit "Unit_1"
    m_SoftwareUnitRelation = m_PlcUnit.Relations.Create("Unit_1",
    UnitRelationType.SoftwareUnit);
    //...
}
```

Ändern Sie folgenden Programmcode, um eine neue Beziehung zu erstellen, indem Sie einen leeren String für den Namen des bezogenen Objekts angeben:

```
private void CreateRelationForRelatedObject(PlcUnit m_PlcUnit)
{
    //...
    m_SoftwareUnitRelation = m_PlcUnit.Relations.Create(string.Empty,
    UnitRelationType.SoftwareUnit);
    //...
}
```

Hinweis

In allen obigen Programmcode-Abschnitten mit Fehlerszenarios wird die Beziehung nicht erstellt und es wird eine wiederherstellbare Ausnahme ausgelöst.

Programmcode: Vorhandene Beziehungen löschen

Sie können eine neue Beziehung erstellen, indem Sie die entsprechende Beziehungsart und den Namen des bezogenen Objekts angeben.

```
private void DeleteExistingRelation(PlcUnitProvider plcUnitProvider, string plcUnitName =
"Unit_2", string dataBlockName = "DB_Global", string technologicalName = "Axis_TO")
{
    private PlcUnitRelation m_SoftwareUnitRelation;
    private PlcUnitRelation m_NonUnitDBRelation;
    private PlcUnitRelation m_TODBRelation;
    // ...
    m_PlcUnit = plcUnitProvider.UnitGroup.Units[0]; //assuming existing units
    m_PlcUnit2 = plcUnitProvider.UnitGroup.Units[1]; //assuming existing units
    m_SoftwareUnitRelation = m_PlcUnit.Relations.Create("Unit_2",
    UnitRelationType.SoftwareUnit);
    m_NonUnitDBRelation = m_PlcUnit.Relations.Create("DB_Global", UnitRelationType.NonUnitDB);
    m_TODBRelation = m_PlcUnit.Relations.Create("Axis_TO", UnitRelationType.TODB);
}
```

Sie können Beziehungen auf mehrere Arten löschen.

Ändern Sie folgenden Programmcode, um die Beziehung zu löschen, auf die der Indexer zugreift:

```
private void DeleteUnitRelationByIndex(PlcUnit m_PlcUnit)
{
    //...
    m_PlcUnit.Relations[0].Delete();
    //...
}
```

Ändern Sie folgenden Programmcode, um die Beziehung direkt zu löschen:

```
private void DeleteUnitRelation(PlcUnitRelation m_SoftwareUnitRelation)
{
    // ...
    m_SoftwareUnitRelation.Delete();
    // ...
}
```

Programmcode: Vorhandene Beziehungen ändern

Sie können Beziehungen auf mehrere Arten ändern:

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Sie können eine neue Beziehung erstellen, indem Sie die entsprechende Beziehungsart und den Namen des bezogenen Objekts angeben.

```
private void ModifyingUnitRelation(PlcUnitProvider plcUnitProvider, string plcUnitName =
"Unit_2")
{
private PlcUnitRelation m_Relation;
m_PlcUnit = plcUnitProvider.UnitGroup.Units[0];
//assuming existing units
m_PlcUnit2 = plcUnitProvider.UnitGroup.Units[1];
//assuming existing units
m_Relation = m_PlcUnit.Relations.Create("Unit_2", UnitRelationType.SoftwareUnit);
// ...
}
```

Ändern Sie folgenden Programmcode, um den Namen des bezogenen Objekts zu aktualisieren, indem Sie einen neuen Wert für das Attribut RelatedObject direkt angeben:

```
private void ModifyingUnitRelation(PlcUnitRelation m_Relation)
{
// ...
m_Relation.RelatedObject = "Unit_3";
// ...
}
```

Ändern Sie folgenden Programmcode, um den Namen des bezogenen Objekts zu aktualisieren, indem Sie einen neuen Wert für das Attribut RelatedObject über SetAttribute zuordnen:

```
private void ModifyingUnitRelation(PlcUnitRelation m_Relation)
{
//...
m_Relation.SetAttribute("RelatedObject", "Unit_4");
//...
}
```

Ändern Sie folgenden Programmcode, um den Namen des bezogenen Objekts zu aktualisieren, indem Sie einen neuen Wert für das Attribut RelatedObject über SetAttributes zuordnen:

```
private void ModifyingUnitRelation(PlcUnitRelation m_Relation)
{
// ...
IList attrList = new List<KeyValuePair<string, object>>();
{
new KeyValuePair<string, object>("RelatedObject", "Unit_4")
};
m_Relation.SetAttributes(attrList);
// ...
}
```

Mögliche Fehlerszenarien beim Ändern vorhandener Beziehungen

Ändern Sie folgenden Programmcode, um eine Beziehung so zu aktualisieren, dass sie auf ein nicht vorhandenes bezogenes Objekt zugreift:

```
private void ModifyingUnitRelation(PlcUnitRelation m_Relation)
{
    //...
    //assuming Plc unit "NonExistingUnit" is not existing in the Plc yet
    m_Relation.RelatedObject = "NonExistingUnit";
    //...
}
```

Hinweis

Durch den obigen Programmcode wird die Beziehung geändert und keine Ausnahme ausgelöst.

Ändern Sie folgenden Programmcode, um eine Beziehung zu aktualisieren, indem Sie einen Namen für das bezogene Objekt angeben, der nicht den TIA-Benennungsregeln entspricht:

```
private void ModifyingUnitRelation(PlcUnitRelation m_Relation)
{
    //...
    m_Relation.RelatedObject = " Unit_2";
    //...
}
```

Ändern Sie folgenden Programmcode, um eine Beziehung so zu aktualisieren, dass sie auf sich selbst zugreift:

```
private void ModifyingUnitRelation(PlcUnitRelation m_Relation)
{
    //...
    //assuming m_Relation is defined under Plc unit "Unit_1"
    m_Relation.RelatedObject = "Unit_1";
    //...
}
```

Hinweis

Durch den obigen Programmcode wird die Beziehung nicht geändert und es wird eine wiederherstellbare Ausnahme aufgrund von führenden Leerzeichen ausgelöst.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

Ändern Sie folgenden Programmcode, um eine Beziehung so zu aktualisieren, dass ein Duplikat zu einer bereits bestehenden Beziehung mit der gleichen Beziehungsart und dem gleichen bezogenen Objekt entsteht:

```
private void ModifyingUnitRelation(PlcUnitRelation m_Relation)
{
    //...
    //assuming a relation with the same relation type and related object is already existing
    m_Relation.RelatedObject = "Unit_2";
    //...
}
```

Ändern Sie folgenden Programmcode, um eine Beziehung zu aktualisieren, indem Sie einen leeren String für den Namen des bezogenen Objekts angeben:

```
private void ModifyingUnitRelation(PlcUnitRelation m_Relation)
{
    //...
    m_Relation.RelatedObject = string.Empty;
    //...
}
```

Hinweis

In allen obigen Programmcode-Abschnitten wird die Beziehung nicht geändert und es wird eine wiederherstellbare Ausnahme ausgelöst.

5.11.6.10 Zugriff auf Namespaces für Software-Einheiten

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Mit TIA Portal Openness können Sie das Namespace-Attribut für Software-Einheiten abrufen und einstellen.

Name der Eigenschaft	Datentyp	Zugriff
NamespacePreset	String	Lesen/Schreiben

Programmcode

Ändern Sie folgenden Programmcode, um das Namespace-Attribut für die Unit festzulegen und abzurufen:

```
private void AccessNamespacesForUnits(PlcUnitProvider unitProvider, string plcUnitName =
"Unit_1")
{
//To set the Namespace attribute
var unit = unitProvider.UnitGroup.Units.Find(plcUnitName);
unit.NamespacePreset = "Siemens.Simatic";
//To get the Namespace attribute
string unitNamespace = unit.NamespacePreset;
}
```

Siehe auch

Projekt öffnen (Seite 131)

5.11.6.11 Zugriff auf Namespaces für Programmbausteine**Voraussetzung**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können TIA Portal Openness verwenden, um das Namespace-Attribut für Programmbausteine unter Software-Einheiten über das Attributset zu vergeben. Der den Programmbausteinen zugeordnete Namespace kann über SIMATIC-ML unter Units exportiert und importiert werden.

Name der Eigenschaft	Datentyp	Zugriff
Namespace	String	Lesen/Schreiben

Programmcode

Ändern Sie den folgenden Programmcode, um das Namespace-Attribut zu setzen und abzurufen:

```
private void static AccessNamespaceForProgramBlock()
var nameSpace = unitBlock.GetAttribute(name:"Namespace");
unitBlock.SetAttribute(name:"Namespace", value:"Unit.Siemens");
```

Namespace-Attribut in SIMATIC-ML

```
</Sections></Interface>
<IsIECCheckEnabled>false</IsIECCheckEnabled>
<Name>Block_1</Name>
<Namespace>Siemens.Simatic</Namespace>
<Number>1</Number>
<ProgrammingLanguage>LAD</ProgrammingLanguage>
<SetENOAutomatically>false</SetENOAutomatically>
<UDABlockProperties/>
<UDAEnableTagReadback>false</UDAEnableTagReadback>
</AttributeList>
<ObjectList>
<MultilingualText ID="1" CompositionName="Comment">
<ObjectList>
<MultilingualTextitem ID="2" CompositionName=":Items">
<AttributeList>
<Culture>en-US</Culture>
<Text/>
</AttributeList>
</MultilingualTextItem>
</ObjectList>
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.11.6.12 Auf Namespaces für UDT zugreifen

Voraussetzung

- Die Anwendung ist über TIA Portal Openness mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie über `GetAttribute` und `SetAttribute` das Namespace-Attribut unter Software-Units den UDTs zuweisen und einstellen. Der den UDTs zugewiesene Namespace kann unter den Units über `SimaticML` exportiert und importiert werden.

Name der Eigenschaft	Datentyp	Zugriff
Namespace	String	Lesen/Schreiben

Programmcode

Ändern Sie den folgenden Programmcode, um das Namespace-Attribut zu setzen und abzurufen:

```
private void AccessNamespaceForUDT()  
{  
    //Namespace can be set and get to the UDTs via the property  
    unitUDT.Namespace = "Motor";  
    var namespaceValue = unitUDT.Namespace;  
    //Namespace can be set and get to the UDTs via the attribute set  
    var namespaceValue = unitUDT.GetAttribute("Namespace");  
}
```

Namespace-Attribut in SIMATIC-ML

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
<Engineering version="V18" />
<DocumentInfo>
<Created>2022-08-03T05:28:40.722228Z</Created>
<ExportSetting>None</ExportSetting>
<InstalledProducts>
<Product>
<DisplayName>Totally Integrated Automation Portal</DisplayName>
<DisplayVersion>V18</DisplayVersion>
</Product>
<OptionPackage>
<DisplayName>TIA Portal Version Control Interface</DisplayName>
<DisplayVersion>V18</DisplayVersion>
</OptionPackage>
<Product>
<DisplayName>Feature Cycle 2 TIA Portal</DisplayName>
<DisplayVersion>V18</DisplayVersion></Product>
<Product>
<DisplayName>Feature Cycle 5 TIA Portal</DisplayName>
<DisplayVersion>V18</DisplayVersion>
</Product>
<Product>
<DisplayName>STEP 7 Professional</DisplayName>
<DisplayVersion>V18</DisplayVersion>
</Product>
<OptionPackage>
<DisplayName>STEP 7 Safety</DisplayName>
<DisplayVersion>V18</DisplayVersion>
</OptionPackage>
</InstalledProducts>
</DocumentInfo>
<SW.Types.PlcStruct ID="0">
<AttributeList>
<Interface><Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v5">
<Section Name="None">
<Member Name="Element_1" Datatype="Bool" />
</Section>
</Sections></Interface>
<Name>UDT_1</Name>
<Namespace>Siemens</Namespace>
</AttributeList>
<ObjectList>
<MultilingualText ID="1" CompositionName="Comment">
<ObjectList>
<MultilingualTextItem ID="2" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
<MultilingualText ID="3" CompositionName="Title">
```

```
<ObjectList>
<MultilingualTextItem ID="4" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
</ObjectList>
</SW.Types.PlcStruct>
</Document>
```

Siehe auch

- Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen (Seite 131)

5.11.6.13 Ladbare Datei für Software Units erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Das TIA Portal-Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)
- Die PLC muss sauber übersetzt sein

Einleitung

Sie können mit TIA Portal Openness über die folgende Methode eine ladbare Datei für Software Units erzeugen: `GenerateLoadable(FileInfo path, IEnumerable<PlcUnit> unit, TargetOption targetOption)`

Hinweis

Das Erzeugen einer ladbaren Datei für Software Units wird bei der PLC1518 oder einem Untertyp und PLCs mit einer Firmware ab V2.8 unterstützt.

Einschränkung

Ladbare Dateien können nur für Software Units erstellt werden, die

- keine Systembausteine enthalten
- keine ProDiag-Bausteine enthalten
- keine fehlersicheren Bausteine enthalten

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

- keine OBs eines anderen Typs als ProgramCycle enthalten
- keinen Baustein mit dynamischer Bindung enthalten
- keine technologischen Objekte enthalten
- keine Benutzer-Datentypen enthalten

Parameter und Enum-Werte

GenerateLoadable() akzeptiert zur Erzeugung einer ladbaren Datei für Software Units folgende Parameter:

Parameterna-me	Datentyp	Beschreibung	Hinweis
path	FileInfo	Gibt Namen und Speicherort der ladbaren Datei an.	Bereits vorhandene ladbare Dateien können nicht ersetzt werden und das Erstellen des übergeordneten Verzeichnisses ist nicht möglich.
units	System.Collections.Generic.IEnumerable<Siemens.Engineering.SW.Units.PlcUnit>	Gibt die Software Units an, die in die ladbare Datei aufgenommen werden sollen.	PlcProgramLoader V1.0 unterstützt nur das Laden einer Software Unit.
targetOption	Siemens.Engineering.SW.Loader.TargetOption	Gibt an, ob die ladbare Datei für eine echte oder simulierte PLC erstellt werden soll.	Wenn TargetOption.PlcSim übergeben wird, müssen alle Bausteine mit Simulationsunterstützung übersetzt worden sein

Der Parameter TargetOption verwendet zur Erzeugung einer ladbaren Datei folgende Enum-Werte:

Wert	Beschreibung
TargetOption.None	Der Standardwert sollte nicht verwendet werden.
TargetOption.Plc	Zur Erzeugung einer ladbaren Datei für eine echte PLC.
TargetOption.PlcSim	Zur Erzeugung einer ladbaren Datei für eine simulierte PLC.

Programmcode

Ändern Sie folgenden Programmcode, um eine ladbare Datei für Software Units zu erzeugen:

```
private void GenerateLoadableFileSoftwareUnit(string loadablePath)
{
    PlcSoftware plc = GetPlc();
    var loadableProviderService = plc.GetService<LoadableProvider>();
    // Create a variable with name "file" to access the loadable path
    var file = new FileInfo(loadablePath);
    // Create a variable with name "unitProvider" to access the unit by calling
    // the GetService() with the PlcUnitProvider type.
    var unitProvider = plc.GetService<PlcUnitProvider>();
    // Create a variable with name "units" for a list of software units.
    var units = new List<PlcUnit>
    {
        // Accessing a specific unit with name "Unit_1"
        unitProvider.UnitGroup.Units.Find("Unit_1") ?? throw new InvalidOperationException()
    };
    // The loadableProviderService generates a loadable file for units by calling
    // the GenerateLoadable() with respective parameters.
    try
    {
        loadableProviderService.GenerateLoadable(file, units, TargetOption.PlcSim);
    }
    // An EngineeringTargetInvocation exception is thrown
    // if generation of the loadable file fails.
    // The exception message contains hints on
    // what caused the loadable file creation to fail.
    catch (EngineeringTargetInvocationException e)
    {
        Console.WriteLine("Generating the loadable file failed.");
        Console.WriteLine(e);
    }
}
```

Ausnahme

Die folgenden möglichen Ausnahmemeldungen sind unten aufgeführt:

- Eine ladbare Datei kann nur aus einer ganzen Software Unit, nicht aus einem einzelnen Baustein innerhalb einer Unit erstellt werden.
- Die ausgewählten Bausteine stammen aus verschiedenen PLCs.
- Zielverzeichnis nicht gefunden: '{0}'.
- Es ist nicht möglich, eine ladbare Datei aus einem fehlersicheren Baustein zu erstellen: '{0}'.
- Zielfeile ist bereits vorhanden: '{0}'.
- Auf die ladbare Datei kann nicht zugegriffen werden.
- Ein interner Fehler ist aufgetreten.
- Die PLC muss übersetzt werden.
- Interne Serialisierung der ladbaren Datei fehlgeschlagen.

5.11 Funktionen für den Zugriff auf die Daten eines PLC-Geräts

- Die interne Serialisierung ist fehlgeschlagen, weil die ladbare Datei entfernt wurde.
- Der angegebene Pfad oder Dateiname oder beides überschreitet die vom System definierte maximale Länge.
- Die PLC '{0}' wird nicht unterstützt.
- Die übergeordnete PLC des Dienstes und des Ziels weichen voneinander ab.
- Ein(ige) Baustein(e) kann (können) nicht simuliert werden. Wenn es sich bei dem Baustein um einen Bibliotheksbaustein handelt, verwenden Sie eine Bibliothek mit Simulationsunterstützung. Andernfalls wählen Sie in den Projekteigenschaften die Option "Beim Übersetzen von Bausteinen Simulierbarkeit unterstützen" ("Support simulation during block compilation") und übersetzen Sie den Baustein erneut.
- Es ist nicht möglich, eine ladbare Datei aus einem Systembaustein zu erstellen: '{0}'.
- Der Zugriff auf die Datei wird verweigert.
- Software Units, die PLC-Datentypen enthalten, werden nicht unterstützt.
- Es ist nicht möglich, eine ladbare Datei aus (einem) ausgewählten Objekt(en) zu erstellen.
- Es ist nicht möglich, eine ladbare Datei aus dem OB '{0}' zu erstellen, weil er nicht die Ereignisklasse 'Programmzyklus' aufweist.
- Es ist nicht möglich, eine ladbare Datei aus einem ProDiag-Baustein zu erstellen: '{0}'.
- Es ist nicht möglich, eine ladbare Datei aus dem ausgewählten Objekt zu erstellen: '{0}'.
- Es ist nicht möglich, eine ladbare Datei aus dem Baustein '{0}' mit aktivierter dynamischer Bindung zu erstellen.

Siehe auch

Ladbare Datei für Bausteine außerhalb von Software Units erzeugen (Seite 543)

5.11.7 Funktionen für fehlersichere Units

5.11.7.1 Fehlersichere Unit generieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness das fehlersichere Programm in Software Units hinzufügen. Die Safety-Software-Unit ist eine vom System generierte Software Unit, die nur auf fehlersicheren PLCs gehostet werden kann. Informationen zu Software Units finden Sie unter Mit Software-Units arbeiten (Seite 617)

Sie können das fehlersichere Programm unter einer Software Unit nur generieren, wenn die Option SafetyInUnit über die API ManagementOfFail-safeInSoftwareUnitsEnvironment aktiviert ist. Sie können eine neue fehlersichere PLC anlegen, die Software Units unterstützt. Es wird dann automatisch eine Safety Unit erstellt, die als einziger Host des F-Programms in der Software-Unit-Umgebung dient.

Methode

TIA Portal Openness stellt die folgenden Methoden für die Generierung einer fehlersicheren PLC mit einem fehlersicheren Programm bereit:

Methode	Beschreibung
bool GenerationOfDefaultFailsafe-Program()	Für den Abruf der Einstellung zum Generieren des fehlersicheren Standardprogramms.
void GenerationOfDefaultFailsafe-Program(bool)	Zum Aktivieren/Deaktivieren der Generierung eines fehlersicheren Standardprogramms beim Erstellen einer neuen fehlersicheren PLC.

TIA Portal Openness stellt die folgenden Methoden für die Generierung eines fehlersicheren Programms unter einer Safety Unit bereit:

Methode	Beschreibung
bool ManagementOfFailsafeInSoftwareUnitsEnvironment()	Für den Abruf des Zustands der jeweiligen Einstellung.
void ManagementOfFailsafeInSoftwareUnitsEnvironment (bool)	Zum Aktivieren/Deaktivieren des Hostings des fehlersicheren Programms in einer Safety Unit.

Programmcode

```

using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
using Siemens.Engineering.SW.Units;
namespace GenerateFailSafeUnit
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            // Generate default fail-safe program under PLC:
            GlobalSettings globalSettingsService = myTiaPortal.GetService<GlobalSettings>();
            // Get the value
            bool isDefaultFailsafeProgram = globalSettingsService.GenerationOfDefaultFailsafeProgram();
            // Set the value
            globalSettingsService.GenerationOfDefaultFailsafeProgram(true);
            // Generate fail-safe program under SafetyUnit environment
            GlobalSettings globalSettingsService = myTiaPortal.GetService<GlobalSettings>();
            // Get the value
            bool isSafetyUnitSupported =
            globalSettingsService.ManagementOfFailsafeInSoftwareUnitsEnvironment();
            // Set the value
            globalSettingsService.ManagementOfFailsafeInSoftwareUnitsEnvironment(true);
        }
    }
}

```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.11.7.2 Auf die Safety Unit zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness über die UnitGroup "PlcSafetyUnitComposition" auf eine Safety Unit zugreifen. Wenn die PLC angelegt ist, kann die Safety Unit nicht mehr erstellt oder gelöscht werden.

Die Safety Unit ist eine vom System generierte Unit.

Attribut

TIA Openness unterstützt die folgende Eigenschaft für den Zugriff auf die Safety Unit, die im Ordner SoftwareUnits im TIA Portal angelegt wurde:

Name der Eigenschaft	DataType	Zugriff
SafetyUnits	PlcSafetyUnitComposition	Lesen/Schreiben

Programmcode

Ändern Sie folgenden Programmcode, um auf die fehlersichere Unit zuzugreifen:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
using Siemens.Engineering.SW.Units;
namespace AccessingSafetyUnit
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            var plcUnitProvider = plcSoftware.GetService<PlcUnitProvider>();
            PlcSafetyUnit safetyUnit = plcUnitProvider.UnitGroup.SafetyUnits.Find("SafetyUnit");
            PlcSafetyUnitComposition plcSafetyUnit = plcUnitProvider.UnitGroup.SafetyUnits; //- Create
            method is not accessible like
            //in standard unit and compile error is thrown
        }
    }
}
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.11.7.3 Beziehungen für die Safety Unit anlegen/löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können mit TIA Portal Openness Beziehungen der Safety Unit erstellen und löschen. Die TOs und GlobalDBs können auf die gleiche Weise wie Beziehungen der Standard-Unit erstellt und gelöscht werden.

Programmcode

Ändern Sie folgenden Programmcode, um die Beziehungen zu erstellen und zu löschen:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
using Siemens.Engineering.SW.Units;
namespace CreateDeleteRelationshipsForTheSafetyUnit
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            // Create relation
            PlcSoftware plcSoftware = ...;
            var plcUnitProvider = plcSoftware.GetService<PlcUnitProvider>();
            var safetyUnit = plcUnitProvider.UnitGroup.SafetyUnits.Find("SafetyUnit");
            var unitRelation = safetyUnit.Relations.Create("Unit_1", UnitRelationType.SoftwareUnit);
            // Delete relation
            unitRelation.Delete();
        }
    }
}
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.11.7.4 Elemente/Objekte einer Safety Unit anlegen bzw. darauf zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können mit TIA Portal Openness die unterlagerten Objekte wie Bausteine, Variablen, UDT der Safety Unit erstellen, löschen, auf diese zugreifen, sie aktualisieren und veröffentlichen. Sie können mit Objekten unter der Safety Unit ebenso wie für die Standard-Unit arbeiten, mit Ausnahme der von einem F-System generierten Bausteine. Sie können ferner die Bausteine unter der Safety Unit mit TIA Portal Openness exportieren und importieren.

Programmcode

Ändern Sie folgenden Programmcode, um Variablen für Safety Units zu erstellen und darauf zuzugreifen:

```
PlcUnitProvider plcUnitProvider = sw.GetService<PlcUnitProvider>();
PlcSafetyUnit safetyUnit = plcUnitProvider.UnitGroup.SafetyUnits.Find("SafetyUnit");
// Create tagtable
var tagTable = safetyUnit.TagTableGroup.TagTables.Create("Tag_Table_1");
// Create tag
var newTag = tagTable.Tags.Create("tag_1");
// Delete tag
newTag.Delete();
// Search tag
var expectedTag = tagTable.Tags.Find("tag_1");
```

Ändern Sie folgenden Programmcode, um Bausteine einer Safety Unit zu exportieren und zu importieren:

```
PlcSafetyUnit safetyUnit = plcUnitProvider.UnitGroup.SafetyUnits.Find("SafetyUnit");
var unitBlock = safetyUnit.BlockGroup.Blocks.Find("Block_2");
// Export the block
unitBlock.Export(new FileInfo(@"D:\Export\SafetyBlock_unit.xml"), ExportOptions.None);
// Import of block
safetyUnit.BlockGroup.Blocks.Import(new
FileInfo( @"C:\Users\z003hzed\Desktop\Block1.xml"), ImportOptions.Override);
```

Programmcode

Ändern Sie folgenden Programmcode:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
using Siemens.Engineering.SW.Units;
namespace CreatingAndAccessingSafetyUnitElement
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            // To create and access safety unit tags
            PlcUnitProvider plcUnitProvider = sw.GetService<PlcUnitProvider>();
            PlcSafetyUnit safetyUnit = plcUnitProvider.UnitGroup.SafetyUnits.Find("SafetyUnit");
            // Create tagtable
            var tagTable = safetyUnit.TagTableGroup.TagTables.Create("Tag_Table_1");
            // Create tag
            var newTag = tagTable.Tags.Create("tag_1");
            // Delete tag
            newTag.Delete();
            // Search tag
            var expectedTag = tagTable.Tags.Find("tag_1");
            // To export and import safety unit blocks
            PlcSafetyUnit safetyUnit = plcUnitProvider.UnitGroup.SafetyUnits.Find("SafetyUnit");
            var unitBlock = safetyUnit.BlockGroup.Blocks.Find("Block_2");
            // Export the block
            unitBlock.Export(new FileInfo(@"D:\Export\SafetyBlock_unit.xml"), ExportOptions.None);
            // Import of block
            safetyUnit.BlockGroup.Blocks.Import(new
            FileInfo( @"C:\Users\z003hzed\Desktop\Block1.xml"), ImportOptions.Override);
        }
    }
}
```

Siehe auch

Projekt öffnen (Seite 131)

5.11.7.5 Überwachungen unter einer Safety Unit exportieren/importieren**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness eine Überwachung unter Safety Units exportieren und importieren, indem auf den Überwachungsprovider der ProDiag-Bausteine der Safety Unit zugegriffen wird.

Die folgenden Einschränkungen gelten für die Safety Unit:

- Eine Safety Unit im Standalone-Betrieb kann nicht über TIA Portal Openness erstellt, gelöscht oder kopiert werden
- Eine Safety Unit kann nicht über TIA Portal Openness herunter- und hochgeladen werden
- Die gehostete Safety Unit kann in TIA Portal Openness nicht in die Masterkopie des Projekts/der globalen Bibliothek kopiert werden

Programmcode

Ändern Sie folgenden Programmcode, um Überwachungen unter einer Safety Unit zu exportieren und zu importieren:


```

using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
using Siemens.Engineering.SW.Units;
namespace ExportingImportingSupervisionsUnderSafetyUnit
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            string filePath = @"E:\Temp\...\fileName.xlsx";
            PlcSafetyUnit safetyUnit = ...;
            // Export from prodiag supervision provider
            var proDiagBlock = safetyUnit.BlockGroup.Blocks.CreateFB("Block_1", true, 1,
                ProgrammingLanguage.ProDiag);
            var proDiagSupervisionProvider = proDiagBlock.GetService<SupervisionProvider>();
            var superVisionResult = proDiagSupervisionProvider.ExportSupervisionsToXlsx(new
                FileInfo(filePath));
            // Import from prodiag supervision provider
            ImportOptions importOptions = ImportOptions.None;
            var superVisionResult = proDiagSupervisionProvider.ImportSupervisionsFromXlsx(new
                FileInfo(filePath), importOptions);
            // Export from global supervision provider.
            var globalSupervisionProvider = safetyUnit.GetService<SupervisionProvider>();
            var superVisionResult = globalSupervisionProvider.ExportSupervisionsToXlsx(new
                FileInfo(filePath));
            // Import from global supervision provider
            var superVisionResult = globalSupervisionProvider.ImportSupervisionsFromXlsx(new
                FileInfo(filePath), importOptions);
            // Import supervision settings
            var superVisionResult = globalSupervisionProvider.ImportSupervisionSettingsFromXlsx(new
                FileInfo(filePath), importOptions);
        }
    }
}

```

Siehe auch

Projekt öffnen (Seite 131)

5.11.7.6 Bausteine unterhalb einer Safety Unit veröffentlichen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness alle Bausteine (FC, FB, DB und IDB) veröffentlichen, die unter dem Programmbausteinordner der Safety Unit angelegt sind. Alle vom System generierten Bausteine in der Safety Unit sind von der Veröffentlichung ausgeschlossen und beim Zugriff wird eine Runtime-Ausnahme gemeldet.

Programmcode

Ändern Sie den folgenden Programmcode, um das veröffentlichte Attribut unter Safety Unit zu setzen:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
using Siemens.Engineering.SW.Units;
namespace PublishingBlocksUnderSafetyUnit
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            PlcSafetyUnit safetyUnit = plcUnitProvider.UnitGroup.SafetyUnits.Find("SafetyUnit");
            var safetyBlock = safetyUnit.BlockGroup.Blocks.Find("Block_2");
            //To set the access
            safetyBlock.SetAttribute("Access", UnitAccessType.Published);
            //To get the access
            var access = safetyBlock.GetAttribute("Access");
        }
    }
}
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.12 Funktionen für den Zugriff auf die Daten eines HMI-Gerätes

5.12.1 Bilder

5.12.1.1 Benutzerdefinierte Bilderordner erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um einen benutzerdefinierten Bilderordner zu erstellen, ändern Sie folgenden Programmcode:

```
//Creates a screen folder
private static void CreateScreenFolder(HmiTarget hmitarget)
{
    ScreenUserFolder myCreatedFolder =
hmitarget.ScreenFolder.Folders.Create("myScreenFolder");
}
```

5.12.1.2 Bild aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Hinweis

Sie können ein Permanentfenster nicht löschen. Ein Permanentfenster ist ein Systembild, das immer vorhanden ist.

Programmcode

Um ein Bild aus einem bestimmten Ordner zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteScreenFromFolder(HmiTarget hmiTarget)
{
    ScreenUserFolder screenUserFolder =
hmiTarget.ScreenFolder.Folders.Find("myScreenFolder");
    ScreenComposition screens = screenUserFolder.Screens;
    Screen screen = screens.Find("myScreenName");
    if (screen != null)
    {
        screen.Delete();
    }
}
```

5.12.1.3 Bildvorlage aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um eine Bildvorlage aus einem bestimmten Ordner zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteScreenTemplateFromFolder(HmiTarget hmiTarget)
{
    string templateName = "MyScreenTemplate";
    ScreenTemplateUserFolder folder =
hmiTarget.ScreenTemplateFolder.Folders.Find("myScreenTemplateFolder");
    ScreenTemplateComposition templates = folder.ScreenTemplates;
    ScreenTemplate template = templates.Find(templateName);
    if (template != null)
    {
        template.Delete();
    }
}
```

5.12.1.4 Alle Bilder aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Hinweis

Sie können ein Permanentfenster nicht löschen. Ein Permanentfenster ist ein Systembild, das immer vorhanden ist.

Programmcode

Um alle Bilder aus einem bestimmten Ordner zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteAllScreensFromFolder(HmiTarget hmitarget)
//Deletes all screens from a user folder or a system folder
{
    ScreenUserFolder folder = hmitarget.ScreenFolder.Folders.Find("myScreenFolder");
    //or ScreenSystemFolder folder = hmitarget.ScreenFolder;
    ScreenComposition screens = folder.Screens;
    List<Screen> list = new List<Screen>();
    foreach(Screen screen in screens)
    {
        list.Add(screen);
    }
    foreach (Screen screen in list)
    {
        screen.Delete();
    }
}
```

5.12.2 Zyklen

5.12.2.1 Zyklus löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Im Projekt ist ein HMI-Gerät vorhanden.

Verwendung

Standardzyklen können Sie nicht löschen.

Sie können anhand der Zusammensetzung im Objektmodell (composition count) des jeweiligen Zyklus ermitteln, ob tatsächlich Zyklen gelöscht wurden. Auf diese Zyklen ist kein Zugriff mehr möglich.

Programmcode

Um einen Zyklus aus einem HMI-Gerät zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteCycle(HmiTarget hmiTarget)
{
    CycleComposition cycles = hmiTarget.Cycles;
    Cycle cycle = cycles.Find("myCycle");
    cycle.Delete();
}
```

5.12.3 Textlisten

5.12.3.1 Textliste löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um eine ausgewählte Textliste und alle zugehörigen Listeneinträge aus einem HMI-Gerät zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteTextList(HmiTarget hmiTarget)
{
    TextListComposition textLists = hmiTarget.TextLists;
    TextList textList = textLists.Find("myTextList");
    textList.Delete();
}
```

5.12.4 Grafiklisten

5.12.4.1 Grafikliste löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um eine ausgewählte Grafikliste und alle zugehörigen Listeneinträge aus einem HMI-Gerät zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteGraphicList(HmiTarget hmiTarget)
{
    GraphicListComposition graphicLists = hmiTarget.GraphicLists;
    GraphicList graphicList = graphicLists.Find("myGraphicList");
    graphicList.Delete();
}
```


5.12.5 Verbindungen

5.12.5.1 Verbindung löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um eine ausgewählte Kommunikationsverbindung aus einem HMI-Gerät zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteConnection(HmiTarget hmiTarget)
{
    ConnectionComposition connections = hmiTarget.Connections;
    Connection connection = connections.Find("HMI_connection_1");
    connection.Delete();
}
```

5.12.6 Variablentabelle

5.12.6.1 Benutzerdefinierte Ordner für HMI-Variablen erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um einen benutzerdefinierten Ordner für HMI-Variablen zu erstellen, ändern Sie folgenden Programmcode:

```
private static void CreateUserFolderForHMITags(HmiTarget hmitarget)
// Creates an HMI tag user folder
{
    TagSystemFolder folder = hmitarget.TagFolder;
    TagUserFolder myCreatedFolder = folder.Folders.Create("MySubFolder");
}
```

5.12.6.2 Variablen einer HMI-Variablentabelle enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um alle Variablen einer HMI-Variablentabelle zu enumerieren, ändern Sie folgenden Programmcode:

```
private static void EnumerateTagsInTagtable(HmiTarget hmitarget)
// //Enumerates all tags of a tag table
{
    TagTable table = hmitarget.TagFolder.TagTables.Find("MyTagtable");
    // Alternatively, you can access the default tag table:
    // TagTable defaulttable = hmitarget.TagFolder.DefaultTagTable;

    TagComposition tagComposition = table.Tags;
    foreach (Tag tag in tagComposition)
    {
        // Add your code here
    }
}
```

5.12.6.3 Einzelne Variable aus einer HMI-Variablentabelle löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um eine bestimmte Variable aus einer HMI-Variablentabelle zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteATag(HmiTarget hmiTarget)
{
    string tagName = "MyTag";
    TagTable defaultTagTable = hmiTarget.TagFolder.DefaultTagTable;
    TagComposition tags = defaultTagTable.Tags;
    Tag tag = tags.Find(tagName);
    tag.Delete();
}
```

5.12.6.4 Variablentabelle aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Im Projekt ist ein HMI-Gerät vorhanden.

Verwendung

Sie können die Standardvariablentabelle nicht löschen

Programmcode

Ändern Sie folgenden Programmcode:

```
// Delete a tag table from a specific folder
private static void DeleteTagTable(HmiTarget hmiTarget)
{
    string tableName = "myTagTable";
    TagSystemFolder tagSystemFolder = hmiTarget.TagFolder;
    TagTableComposition tagTables = tagSystemFolder.TagTables;
    TagTable tagTable = tagTables.Find(tableName);
    tagTable.Delete();
}
```

5.12.7 VB-Skripte

5.12.7.1 Benutzerdefinierte Ordner für Skripte erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um einen benutzerdefinierten Unterordner für Skripte in einem Systemordner oder einem anderen benutzerdefinierten Ordner zu erstellen, ändern Sie folgenden Programmcode:

```
private static void CreateFolderInScriptfolder(HmiTarget hmitarget)
//Creates a script user subfolderVBScriptSystemFolder
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptUserFolderComposition vbScriptFolders = vbScriptFolder.Folders;
    VBScriptUserFolder vbScriptSubFolder = vbScriptFolder.Folders.Create("mySubfolder");
}
```

5.12.7.2 VB-Skript aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um ein VB-Skript aus einem bestimmten Ordner zu löschen, ändern Sie folgenden Programmcode:

```
//Deletes a vbscript from a script folderVBScriptSystemFolder
private static void DeleteVBScriptFromScriptFolder(HmiTarget hmitarget)
{
    VBScriptUserFolder vbscriptfolder =
hmitarget.VBScriptFolder.Folders.Find("MyScriptFolder");
    var vbScripts = vbscriptfolder.VBScripts;
    if (null != vbScripts)
    {
        var vbScript = vbScripts.Find("MyScript");
        vbScript.Delete();
    }
}
```

5.12.7.3 Benutzerdefinierten Ordner eines Bediengeräts löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie den folgenden Programmcode, um einen benutzerdefinierten Ordner eines Bediengeräts zu löschen:

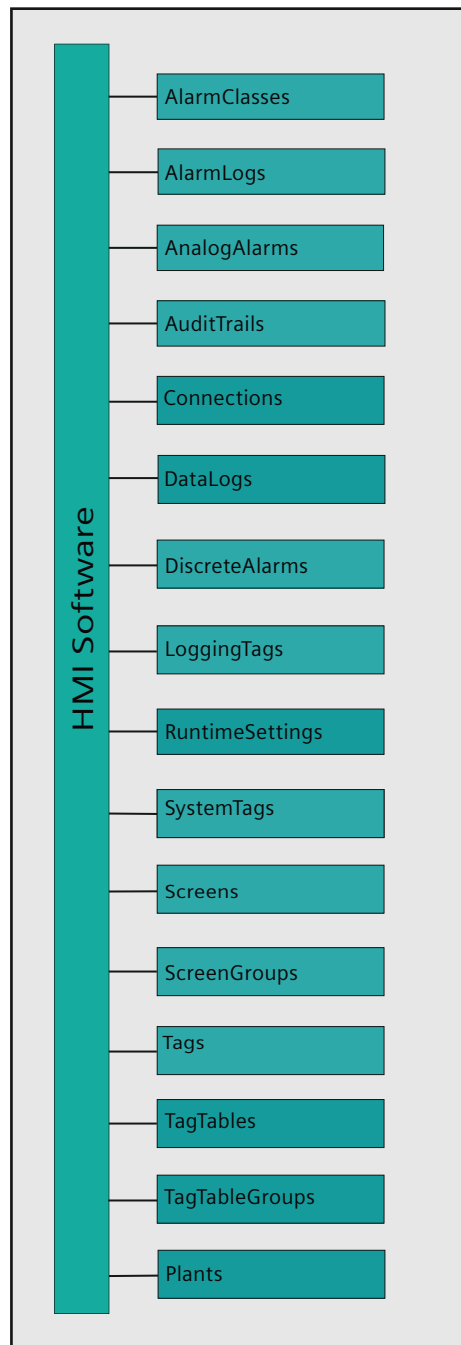
```
HmiTarget hmiTarget = ...;
ScreenUserFolder screenUserGroup = hmiTarget.ScreenFolder.Folders.Find("MyUserFolder");
screenUserGroup.Delete();
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

5.13.1 HMISoftware (RT Unified)

5.13.1.1 HMI-Softwareobjekt (RT Unified)

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)



5.13.1.2 Beschreibung HMISoftware (RT Unified)

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Öffnen eines Projekts (Seite 131)

Einleitung

Sie können das HMI-Softwareobjekt verwenden, um auf Elemente wie Variablen, Verbindungen, Meldungen und Bilder zuzugreifen.

Programmcode

Um auf das HMI-Softwareobjekt zuzugreifen, ändern Sie den folgenden Programmcode:

```
using System;
using System.IO;
using Siemens.Engineering;
using Siemens.Engineering.HmiUnified;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
private HmiSoftware GetHmiSoftware()
{
    HmiSoftware hmiSoftware = null;
    Project project = null;
    IList<TiaPortalProcess> tiaProcessList = TiaPortal.GetProcesses();
    //If no TIA Application instance is running, then following using statement will start it or
    //If TIA application is already running then will attach to it.
    TiaPortal tiaApp = tiaProcessList.Count > 0? tiaProcessList[0].Attach(): new
    TiaPortal(TiaPortalMode.WithUserInterface);
    // tiaApp.Projects[X] defines the project which has to be opened
    // With order number 0 the first project will be opened if several projects are present in
    // given TIA instance.
    if (tiaApp.Projects.Count > 0)
    project = tiaApp.Projects[0];
    else
    {
        //If there is no device project in given TIA instance, then open the existing project.
        FileInfo file = new FileInfo(@"D:\Automation\Project1\Project1.apx");
        if (!file.Exists)
        throw new FileNotFoundException("Project1.apx not found");
        project = tiaApp.Projects?.Open(file);
    }
    //After getting project, get the object representing UA software HMI device.
    var devices = project.Devices;
    if (devices != null)
    {
        //If the device is in the first position of the openness project tree
        var device = devices[0];
        //If the device is not in the first position of the openness project tree, you have to adapt
        // the order number devices[X];
        var deviceItems = device.DeviceItems;
        if (deviceItems != null)
        {
            foreach (DeviceItem deviceItem in deviceItems)
            {
                SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
                hmiSoftware = softwareContainer?.Software as HmiSoftware;
            }
        }
    }
    return hmiSoftware;
}
```

Hinweis

Im obigen Programmcode bezieht sich die Erweiterung .apx auf die installierte Version des TIA Portals.

5.13.1.3 AlarmClasses (RT Unified)**Beschreibung AlarmClasses (RT Unified)****Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um an Meldeklassen eine Aktivität auszuführen, können Sie entweder die Meldeklassen enumerieren oder Find() verwenden.

```
private void BrowseAlarmClasses()
{
    //User can navigate all alarm classes of device
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAlarmClassComposition alarmClasses = hmiSoftware.AlarmClasses;
    foreach (HmiAlarmClass alarmClass in alarmClasses)
    {
        //Working with alarm classes
    }
}
```

Um anhand des Namens auf eine einzelne Meldeklasse zuzugreifen, ändern Sie folgenden Programmcode:

```
private void SearchAlarmClasses()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for a specific alarm class from list of alarm classes
    string alarmClassName = "AlarmClass_1";
    HmiAlarmClassComposition alarmClasses = hmiSoftware.AlarmClasses;
    HmiAlarmClass alarmClassObj = alarmClasses.Find(alarmClassName);
}
```

Mit Meldeklassen arbeiten

Sie können, während Sie TIA Portal Openness verwenden, die folgenden Aktivitäten an Variablen ausführen:

- Meldeklassen erstellen: AlarmClasses.Create() (Seite 679)
- Meldeklassen löschen AlarmClasses.Delete() (Seite 680)
- Eigenschaften von Meldeklassen: Auf Eigenschaften von Meldeklassen zugreifen (Seite 680)
- Querverweisdienst für Meldeklassen: Über Meldeklassen auf den Querverweisdienst zugreifen (Seite 682)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

AlarmClasses.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Meldeklasse mit der Methode Create() der Klasse HmiAlarmClassComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void AlarmClassCreate()
{
    //Create Alarm Class with name "AlarmClass1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAlarmClassComposition alarmClasses = hmiSoftware.AlarmClasses;
    HmiAlarmClass alarmClass = alarmClasses.Create("AlarmClass1");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

AlarmClasses.Delete() (RT Unified)**Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um eine Meldeklasse mit der Methode Delete() der Klasse HmiAlarmClass zu löschen, ändern Sie folgenden Programmcode:

```
public void DeleteAlarmClass()
//User wants to delete an alarm class with the alarm name AlarmClass1
{
HmiSoftware hmiSoftware = GetHmiSoftware();
HmiAlarmClassComposition alarmClasses = hmiSoftware.AlarmClasses;;
HmiAlarmClass alarmClass = alarmClasses.Find("AlarmClass1");
alarmClass.Delete();
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Auf Eigenschaften von Meldeklassen zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden in Meldeklassen unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Name	String	Gibt den Namen einer Meldeklasse an.	Abhängig von der Art der angegebenen Meldeklasse (systembasierte Meldung oder benutzerbasierte Meldung)
Priority	Byte	Gibt die Priorität einer Meldeklasse an.	L/S
IsSystem	Boolean	Gibt an, ob die Meldeklasse vom System vorgegeben wird.	R
StateMachine	HmiAlarmStateMachine	Gibt die Ablaufsteuerung einer Meldeklasse an.	Abhängig von der Art der angegebenen Meldeklasse (systembasierte Meldung oder benutzerbasierte Meldung)
RaisedState	HmiRaisedState	Gibt den Zustand für aktive Meldungen an.	R

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
RaisedState.BackColor	Color	Gibt die Hintergrundfarbe für den aktiven Zustand an.	L/S
RaisedState.TextColor	Color	Gibt die Textfarbe für den aktiven Zustand an.	L/S
RaisedState.Flashing	Boolean	Gibt das Blinken für den aktiven Zustand an.	L/S
ClearedState	HmiClearedState	Gibt den Zustand für gelöschte Meldungen an.	R
ClearedState.BackColor	Color	Gibt die Hintergrundfarbe für den gelöschten Zustand an.	L/S
ClearedState.TextColor	Color	Gibt die Textfarbe für den gelöschten Zustand an.	L/S
ClearedState.Flashing	Boolean	Gibt das Blinken für den gelöschten Zustand an.	L/S
AcknowledgedState	HmiAcknowledgedState	Gibt den Zustand für quittierte Meldungen an.	R
AcknowledgedState.BackColor	Color	Gibt die Hintergrundfarbe für den quittierten Zustand an.	L/S
AcknowledgedState.TextColor	Color	Gibt die Textfarbe für den quittierten Zustand an.	L/S
AcknowledgedState.Flashing	Boolean	Gibt das Blinken für den quittierten Zustand an.	L/S
AcknowledgedClearedState	HmisAcknowledgedClearedState	Gibt den Zustand für quittierte und gelöschte Meldungen an.	-
AcknowledgedClearedState.BackColor	Color	Gibt die Hintergrundfarbe für quittierte und gelöschte Meldungen an.	L/S
AcknowledgedClearedState.TextColor	Color	Gibt die Textfarbe für quittierte und gelöschte Meldungen an.	L/S
AcknowledgedClearedState.Flashing	Boolean	Gibt das Blinken für quittierte und gelöschte Meldungen an.	L/S
Id	UInt32	Gibt die ID zum Identifizieren der Meldeklasse an. Diese ID ist ein eindeutiger Wert für jede Meldeklasse.	R
Log	System.String	Gibt das Archiv einer Meldeklasse an.	L/S
CommonAlarmClass	System.String	Gibt die gemeinsame Meldeklasse an.	R

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften von Meldeklassen zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void AlarmClassesPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAlarmClassComposition alarmClasses = hmiSoftware.AlarmClasses;
    HmiAlarmClass alarmClass = hmiSoftware.AlarmClasses.Find("MostCritical");
    int priority = alarmClass.Priority;
    alarmClass.Priority = 10;
    alarmClass.Acknowledgement = StateMachine.AlarmWithDualModeAcknowledgement;
    alarmClass.IncomingOutgoingAcknowledgeStatus.BackColor = Color.Red;
    alarmClass.IncomingOutgoingAcknowledgeStatus.TextColor = Color.Black;
    alarmClass.IncomingOutgoingAcknowledgeStatus.Flashing = true;
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

Über Meldeklassen auf den Querverweisdienst zugreifen (RT Unified)

Voraussetzungen

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um mit der Methode `GetService()` der Klasse `HmiAlarmClass` auf den Querverweisdienst zuzugreifen, ändern Sie folgenden Programmcode:

```
private void getCrossReferenceServiceforTag()
{
    SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
    HmiSoftware hmiSoftware = softwareContainer.Software as HmiSoftware;
    // Accessing lower
    // levels until we reach the HmiAlarmClassComposition
    HmiAlarmClassComposition alarmClasses = hmiSoftware.AlarmClasses;
    HmiAlarmClass alarmClass = alarmClasses.Find("AlarmClass1");
    if (alarmClass != null)
    {
        try
        {
            CrossReferenceService crossReferenceService =
            alarmClass.GetService<CrossReferenceService>();
            .....
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

Siehe auch

HMI-Softwareobjekt (Seite 674)

Projekt öffnen (Seite 131)

Verbindung zum TIA Portal aufbauen (Seite 84)

5.13.1.4 AlarmLogs (RT Unified)**Beschreibung AlarmLogs (RT Unified)****Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung `HmiSoftware` (Seite 676)

Programmcode

Um an Meldearchiven eine Aktivität auszuführen, können Sie entweder die Meldearchive enumerieren oder Find() verwenden.

```
private void AlarmLogBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all the alarm logs of given alarm log table
    HmiAlarmLogComposition alarmLogs = hmiSoftware.AlarmLogs;
    foreach (HmiAlarmLog alarmLog in alarmLogs)
    {
        //working with alarm log.
    }
}
```

Um anhand des Namens auf ein einzelnes Meldearchiv zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SearchAlarmlog()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for a specific alarm log from list of alarmlogs
    String alarmLogName = "AlarmLog_1";
    HmiAlarmLogComposition alarmLogs = hmiSoftware.AlarmLogs;
    HmiAlarmLog alarmLogObj = alarmLogs.Find(alarmLogName);
}
```

Arbeiten mit Meldearchiven

Mit Datenarchiven können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Meldearchive erstellen: AlarmLogs.Create() (Seite 684)
- Meldearchive löschen: AlarmLogs.Delete() (Seite 685)
- Eigenschaften von Meldearchiven: Auf Eigenschaften von Meldearchiven zugreifen (Seite 686)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

AlarmLogs.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um ein Meldearchiv mit der Methode Create() der Klasse HmiAlarmLogComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void AlarmLogCreate()
{
    //Create Alarmlog
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAlarmLogComposition alarmLogs = hmiSoftware.AlarmLogs;
    HmiAlarmLog alarmLog = alarmLogs.Create("Alarmlog1");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

AlarmLogs.Delete() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um ein Meldearchiv mit der Methode Delete() der Klasse HmiAlarmLog zu löschen, ändern Sie folgenden Programmcode:

```
private void AlarmLogDelete()
{
    //User wants to delete alarm log with name "Alarmlog1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAlarmLogComposition alarmLogs = hmiSoftware.AlarmLogs;
    HmiAlarmLog alarmLog = alarmLogs.Find("Alarmlog1");
    alarmLog.Delete();
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

Auf Eigenschaften von Meldearchiven zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in den Meldearchiven unterstützt:

Name der Eigenschaft		Datentyp	Beschreibung	Zugriff
Name		String	Gibt den Namen des Meldearchivs an.	L/S
Settings		LogSettings	Gibt die Mitglieder in komplexen Eigenschaften wie StorageFolder, LogMaxSize und LogTimePeriod an, die die Einstellungen für das Meldearchiv festlegen.	R
	StoragePath	String	Gibt den Pfad für die Archivierung an.	L/S
	LogMaxSize	Unsigned int	Gibt die maximale Größe des Archivs auf dem Speichermedium in Megabyte an.	L/S
	LogTimePeriod	LogDuration	Gibt den vom Meldearchiv abgedeckten maximalen Zeitraum an.	L/S
Segment		LogSegment	Gibt die Mitglieder in komplexen Eigenschaften wie SegmentMaxSize, StartTime und SegmentTimePeriod an, die die Einstellungen für das Segment festlegen.	R
	SegmentMaxSize	Unsigned int	Gibt die maximale Größe des Meldearchivsegments in Megabyte an.	L/S
	SegmentStartTime	DateTime	Gibt den genauen Zeitpunkt an, wenn das Segment gestartet wird.	L/S
	SegmentTimePeriod	SegmentDuration	Gibt den vom Meldearchivsegment abgedeckten maximalen Zeitraum an.	L/S
Backup		LogBackup	Gibt die Mitglieder komplexer Eigenschaften wie PrimaryPath und BackupMode an, die die Einstellungen für das Archiv-Backup festlegen.	R
	BackupMode	HmiBackupMode	Gibt den Backup-Modus an.	L/S
	PrimaryPath	String	Gibt den Pfad für das Backup an.	L/S

Die folgenden Eigenschaften werden in der Archivdauer unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Days	Unsigned int	Gibt die festgelegte und abgerufene Anzahl von Tagen an.	L/S
Hours	Unsigned int	Gibt die festgelegte und abgerufene Anzahl von Stunden an.	L/S
Segment	LogSegment	Gibt die Mitglieder komplexer Eigenschaften wie SegmentMaxSize, StartTime und SegmentTimePeriod an, die die Einstellungen für das Segment festlegen.	R
Backup	LogBackup	Gibt die Mitglieder komplexer Eigenschaften wie PrimaryPath und BackupMode an, die die Einstellungen für das Archiv-Backup festlegen.	R

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften von Meldearchiven zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void AlarmLogPropertiesAccess()
{
    //Set and Get alarm log properties.
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAlarmLogComposition alarmLogs = hmiSoftware.AlarmLogs;
    HmiAlarmLog alarmLog = alarmLogs.Find("Alarmlog_1");
    string name = alarmLog.Name;
    alarmLog.Settings.StorageFolder = @"C:\Logs\Log";
    alarmLog.Settings.LogMaxSize = Int32.MaxValue;
    alarmLog.Settings.LogMaxSize = 2000;
    LogDuration duration = alarmLog.Settings.LogTimePeriod;
    double durationInDouble = duration.GetDoubleLogDuration();
    string durationInString = duration.GetStringLogDuration();
    duration.Days = 7;
    duration.Hours = 23;
    duration.Minutes = 50;
    duration.Seconds = 0;
    duration.Ticks = 1;
    //Log duration can be set by using function also.
    duration.SetLogDuration(10, 12, 4, 5, 0);
    alarmLog.Backup.BackupMode = BackupMode.NoBackup;
    alarmLog.Backup.BackupMode = BackupMode.PrimaryPath;
    alarmLog.Backup.PrimaryPath = @"C:\Logs\Backup";
    alarmLog.Segment.SegmentMaxSize = 500;
    alarmLog.Segment.SegmentStartTime = DateTime.Now;
    alarmLog.Segment.SegmentStartTime = DateTime.Now.Date;
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.5 AnalogAlarms (RT Unified)**Beschreibung AnalogAlarms (RT Unified)****Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um an Analogmeldungen eine Aktivität auszuführen, können Sie entweder die Analogmeldungen enumerieren oder Find() verwenden.

```
private void AnalogAlarmBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all analog alarms of device
    HmiAnalogAlarmComposition analogAlarms = hmiSoftware.AnalogAlarms;
    foreach (HmiAnalogAlarm analogAlarm in analogAlarms)
    {
        //Working with analog alarms.
    }
}
```

Um anhand des Namens auf eine einzelne Analogmeldung zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SearchAnalogAlarm()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for a specific analog alarm from list of analog alarms
    String analogAlarmName = "AnalogAlarm_1";
    HmiAnalogAlarmComposition analogAlarms = hmiSoftware.AnalogAlarms;
    HmiAnalogAlarm analogAlarmObj = analogAlarms.Find(analogAlarmName);
}
```

Mit Analogmeldungen arbeiten

Mit Analogmeldungen können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Variablen erstellen: AnalogAlarms.Create() (Seite 689)
- Variablen löschen: AnalogAlarms.Delete() (Seite 689)
- Eigenschaften von Analogmeldungen: Auf Eigenschaften von Analogmeldungen zugreifen (Seite 690)
- Querverweisdienst für Analogmeldungen: Über Analogmeldungen auf den Querverweisdienst zugreifen (Seite 692)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

AnalogAlarms.Create() (RT Unified)

Voraussetzungen

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Analogmeldung mit der Methode Create() der Klasse HmiAnalogAlarmComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void AnalogAlarmCreate()
{
    //Create an Analog Alarm with name "Alarm1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAnalogAlarmComposition analogAlarms = hmiSoftware.AnalogAlarms;
    HmiAnalogAlarm analogAlarm = analogAlarms.Create("Alarm1");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

AnalogAlarms.Delete() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Analogmeldung mit der Methode Delete() der Klasse HmiAnalogAlarm zu löschen, ändern Sie folgenden Programmcode:

```
private void AnalogAlarmDelete()
{
    //Delete an Analog Alarm
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAnalogAlarmComposition analogAlarms = hmiSoftware.AnalogAlarms;
    HmiAnalogAlarm analogAlarm = analogAlarms.Find("Alarm_1");
    if (analogAlarm != null)
    {
        analogAlarm.Delete();
    }
}
```

Siehe auch

[Verbindung zum TIA Portal aufbauen \(Seite 84\)](#)

[Projekt öffnen \(Seite 131\)](#)

Auf Eigenschaften von Analogmeldungen zugreifen (RT Unified)

Eigenschaft

Die folgende Eigenschaften werden in Analogmeldungen unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
EventText	MultilingualText	Gibt den Meldetext einer Analogmeldung an.	L/S
Id	UInt32	Gibt die ID einer Meldung an. Diese ID ist ein eindeutiger Wert für jede Analogmeldung.	L/S
AlarmClass	String	Gibt die Klasse einer Analogmeldung an.	L/S
Name	String	Gibt den Namen einer Analogmeldung an.	L/S
Priority	Byte	Gibt die Priorität einer Analogmeldung an.	L/S
Origin	String	Gibt den Ursprung einer Analogmeldung an.	L/S
Area	String	Gibt den Bereich einer Analogmeldung an.	L/S
RaisedState-Tag	String	Gibt die Variable an, die die Analogmeldung auslöst.	L/S
ConditionValue	Object supported type: Double, Int16, UInt16, Int32, UInt32, Int64, UInt64, Byte	Gibt den Grenzwert für die Analogmeldung an, der als konstanter Wert angegeben werden kann.	L/S
Condition	HmiAlarmCondition	Gibt den Grenzmodus einer Analogmeldung an.	L/S
InfoText	MultilingualText	Gibt den Infotext einer Analogmeldung an.	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
AlarmParameterTags	Object Supported-Type: IList<string>	Ein Array aus maximal 10 Elementen gibt die Meldeparameter in einem Meldetext an. Jedes Element des Array enthält einen Variablennamen als Zeichenkette.	L/S
EventText1	MultilingualText	Gibt den zusätzlichen Text 1 einer Analogmeldung an.	L/S
EventText2	MultilingualText	Gibt den zusätzlichen Text 2 einer Analogmeldung an.	L/S
EventText3	MultilingualText	Gibt den zusätzlichen Text 3 einer Analogmeldung an.	L/S
EventText4	MultilingualText	Gibt den zusätzlichen Text 4 einer Analogmeldung an.	L/S
EventText5	MultilingualText	Gibt den zusätzlichen Text 5 einer Analogmeldung an.	L/S
EventText6	MultilingualText	Gibt den zusätzlichen Text 6 einer Analogmeldung an.	L/S
EventText7	MultilingualText	Gibt den zusätzlichen Text 7 einer Analogmeldung an.	L/S
EventText8	MultilingualText	Gibt den zusätzlichen Text 8 einer Analogmeldung an.	L/S
EventText9	MultilingualText	Gibt den zusätzlichen Text 9 einer Analogmeldung an.	L/S
TriggerAddress	String	Gibt die Adresse der Auslösevariable an.	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften einer Analogmeldung zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void AnalogAlarmPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAnalogAlarmComposition analogAlarms = hmiSoftware.AnalogAlarms;
    HmiAnalogAlarm analogAlarm = analogAlarms.Find("Alarm_1");
    uint id = analogAlarm.Id;
    analogAlarm.Id = 10;
    int priority = analogAlarm.Priority;
    analogAlarm.Priority = 7;
    string area = analogAlarm.Area;
    analogAlarm.Area = "Area_1";
    string name = analogAlarm.Name;
    analogAlarm.Name = "NewAlarm";
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Über Analogmeldungen auf den Querverweisdienst zugreifen (RT Unified)

Voraussetzungen

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um mit der Methode `GetService()` der Klasse `HmiAnalogAlarm` auf den Querverweisdienst zuzugreifen, ändern Sie folgenden Programmcode:

```
private void getCrossReferenceServiceforAnalogAlarm()
{
    SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
    HmiSoftware hmiSoftware = softwareContainer.Software as HmiSoftware;
    // Accessing lower levels
    // until we reach the HmiAnalogAlarmComposition.
    HmiAnalogAlarmComposition analogAlarms = hmiSoftware.AnalogAlarms;
    HmiAnalogAlarm analogAlarm = analogAlarms.Find("Alarm1")
    if (analogAlarm != null)
    {
        try
        {
            CrossReferenceService crossReferenceService =
            analogAlarm.GetService<CrossReferenceService>();
            .....
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.6 AuditTrails (RT Unified)

Beschreibung Audit Trails (RT Unified)

Voraussetzungen

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung `HmiSoftware` (Seite 676)

Programmcode

Um an Audit Trails eine Aktivität auszuführen, können Sie entweder die Audit Trails enumerieren oder Find() verwenden.

```
private void AuditTrailBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all the audit trails
    HmiAuditTrailComposition auditLogs = hmiSoftware.AuditTrails
    foreach (HmiAuditTrail auditLog in auditLogs)
    {
        //working with audit trails.
    }
}
```

Um anhand des Namens auf einen einzelnen Audit Trail zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SearchAuditTrails()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for specific audit trail name
    string auditTrailName = "AuditTrail_1";
    HmiAuditTrailComposition auditLogs = hmiSoftware.AuditTrails;
    HmiAuditTrail auditLogObject = auditLogs [01];
    AuditTrail auditTrailObj = auditLogObject.Find(auditTrailName);
}
```

Arbeiten mit Audit Trails

Sie können, während Sie TIA Portal Openness verwenden, die folgenden Aktivitäten an Audit Trails ausführen:

- Audit-Trail-Eigenschaften: Auf Eigenschaften von Audit Trails zugreifen (Seite 695)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Auf Eigenschaften von Audit Trails zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in den Audit Trails unterstützt:

Name der Eigenschaft		Datentyp	Beschreibung	Zugriff
Name		String	Gibt den Namen des Audit Trail an.	L/S
Settings		LogSettings	Gibt die Mitglieder komplexer Eigenschaften wie StorageFolder, LogMaxSize und LogTimePeriod an, die die Einstellungen für den Audit Trail festlegt.	R
	StorageFolder	String	Gibt den Ordnerpfad für die Archivierung an.	L/S
	LogMaxSize	Unsigned int	Gibt die maximale Größe des Archivs auf dem Speichermedium in Megabyte an.	L/S
	LogTimePeriod	LogDuration	Gibt den vom Archiv abgedeckten maximalen Zeitraum an.	L/S
	StorageDevice	DeviceNode	Gibt das Speichergerät an.	L/S
Segment		LogSegment	Gibt die Mitglieder komplexer Eigenschaften wie SegmentMaxSize, StartTime und SegmentTimePeriod an, die die Einstellungen für das Segment festlegen.	R
	SegmentMaxSize	Unsigned int	Gibt die maximale Größe des Segments im Audit Trail in Megabyte an.	L/S
	SegmentStartTime	DateTime	Gibt den genauen Zeitpunkt an, wenn das Segment gestartet wird.	L/S
	SegmentTimePeriod	SegmentDuration	Gibt den vom Audit Trail-Segment abgedeckten maximalen Zeitraum an.	L/S
Backup		LogBackup	Gibt die Mitglieder komplexer Eigenschaften wie PrimaryPath und BackupMode an, die die Einstellungen für das Audit Trail-Backup festlegen.	R
	BackupMode	HmiBackupMode	Gibt den Backup-Modus an.	L/S
	PrimaryPath	String	Gibt den primären Pfad für das Backup an.	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften von Audit Trails zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void AuditTrailPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiAuditTrailComposition auditLogs = hmiSoftware.AuditTrails;
    HmiAuditTrail auditLogObject = auditLogs [01];
    string auditTrailName = auditLogObject.Name;
    Console.WriteLine("Name {0}" auditTrailName);
    auditLogObject.Name = "Audit Log_100";
    auditLogObject.Settings.StorageFolder = @"C:\\Tests123";
    auditLogObject.Settings.LogMaxSize = Int32.MaxValue;
    auditLogObject.Settings.LogMaxSize = 2000;
    LogDuration duration = auditLogObject.Settings.LogTimePeriod;
    double durationInDouble = duration.GetDoubleLogDuration();
    string durationInString = duration .GetStringLogDuration();
    duration.Days = 200;
    duration.Hours = 10;
    duration.Minutes = 22;
    duration.Seconds = 39;
    duration.Ticks = 0;
    //Log duration can be set by function also
    duration.SetLogDuration(10,12,4,5,0);
    auditLogObject.Backup.BackupMode = HmiBackupMode.NoBackup;
    auditLogObject.Backup.BackupMode = HmiBackupMode.PrimaryPath;
    auditLogObject.Backup.PrimaryPath = @"C:\Logs\Backup";
    auditLogObject.Segment.SegmentMaxSize = 500;
    auditLogObject.Segment.SegmentStartTime = DateTime.Now;
    auditLogObject.Segment.SegmentStartTime = DateTime.Now.Date;
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.7 Verbindungen (RT Unified)

Beschreibung Verbindungen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um an einer Verbindung eine Aktivität auszuführen, können Sie entweder die Verbindungen enumerieren oder Find() verwenden.

```
private void ConnectionBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all connections of device
    HmiConnectionComposition connections = hmiSoftware.Connections;
    foreach (HmiConnection connection in connections)
    {
        //working with connection
    }
}
```

Um anhand des Namens auf eine einzelne Verbindung zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void ConnectionSearch()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for specific connection name from a list of connections
    string connectionName = "Connection_1";
    HmiConnectionComposition connections = hmiSoftware.Connections;
    HmiConnection connectionObj = connections.Find(connectionName);
}
```

Mit Zusammensetzungen arbeiten

Mit Verbindungen können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Verbindungen erstellen: Connections.Create() (Seite 698)
- Verbindungen löschen: Connections.Delete() (Seite 698)
- Verbindungseigenschaften: Auf Eigenschaften von Verbindungen zugreifen (Seite 699)
- Querverweisdienst für Verbindungen: Über Verbindungen auf den Querverweisdienst zugreifen (Seite 700)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Connections.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um eine Verbindung mit der Methode Create() der Klasse HmiConnectionComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void ConnectionCreate()
{
    //Create a connection with name "Connection_1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiConnectionComposition connections = hmiSoftware.Connections;
    HmiConnection connection = connections.Create("Connection_1");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Connections.Delete() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um eine Verbindung mit der Methode Delete() der Klasse HmiConnection zu löschen, ändern Sie folgenden Programmcode

```
private void ConnectionDelete()
{
    //Delete a connection with name "Connection_1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiConnectionComposition connections = hmiSoftware.Connections;
    HmiConnection connection = connections.Find("Connection_1");
    connection.Delete();
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Auf Eigenschaften von Verbindungen zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden in einer Verbindung unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Name	String	Gibt den Namen der Verbindung an.	R/W
DisabledAtStartup	Boolean	Gibt an, ob die Verbindung beim Start der Runtime deaktiviert ist.	R/W
CommunicationDriver	String	Gibt den Kommunikations-treiber an.	R/W
Node	String	Gibt den Zugangspunkt des Partners (z.B. PLC) an.	R
Partner	String	Gibt den Namen der verbundenen PLC an.	R
Station	String	Gibt den Namen der Station an, der die PLC zugeordnet ist.	R
Comment	String	Gibt gegebenenfalls zusätzliche Kommentare an.	R/W
InitialAddress	String	Gibt die Parameter der Verbindung wie Schnittstellentyp (DP, TCP/IP), IP-Adresse, Baugruppenträger usw. an.	R/W

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften von Verbindungen zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void ConnectionPropertiesAccess()
{
    // Accessing properties of Connection object.
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiConnectionComposition connections = hmiSoftware.Connections;
    HmiConnection connection = connections.Find("Connection_12");
    string name = connection.Name;
    connection.Name = "Connection_15";
    bool online = connection.DisabledAtStartup;
    connection.DisabledAtStartup = true;
    string node = connection.Node;
    string station = connection.Station;
    string partner = connection.Partner;
    string communicationDriver = connection.CommunicationDriver;
    connection.CommunicationDriver = "SIMATIC S7 300/400";
    // Accessing "InitialAddress" property of connection object
    // with valid strings
    connection.InitialAddress = "CommunicationInterface = Industrial Ethernet;
    HostAddress = 127.157.8.1; HostAccessPoint = S7ONLINE; PlcRack = 5; PlcIsCyclicOperation =
    true";
    // Accessing "InitialAddress" property of connection object with invalid value or invalid
    // format will return recoverable exception.
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Über Verbindungen auf den Querverweisdienst zugreifen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um mit der Methode `GetService()` der Klasse `HmiConnection` auf den Querverweisdienst zuzugreifen, ändern Sie folgenden Programmcode:

```
private void getCrossReferenceServiceforConnection()
{
    SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
    HmiSoftware hmiSoftware = softwareContainer.Software as HmiSoftware; //Accessing lower
    levels
    // until we reach the HmiConnectionComposition.
    HmiConnectionComposition connections = hmiSoftware.Connections;
    HmiConnection hmiConnection = connections.Find("Connection1");
    if (hmiConnection != null)
    {
        try
        {
            CrossReferenceService crossReferenceService =
            hmiConnection.GetService<CrossReferenceService>();
            .....
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.8 DataLogs (RT Unified)**Beschreibung AnalogAlarms (RT Unified)****Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung `HmiSoftware` (Seite 676)

Programmcode

Um an Datenarchiven eine Aktivität auszuführen, können Sie entweder die Datenarchive enumerieren oder Find() verwenden.

```
private void DatalogBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all data logs of device
    HmiDataLogComposition dataLogs = hmiSoftware.DataLogs;
    foreach (HmiDataLog dataLog in dataLogs)
    {
        // Working with data logs
    }
}
```

Um anhand des Namens auf ein einzelnes Datenarchiv zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SearchDatalog()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for a specific data log from list of data logs
    String datalogName = "DataLog_1";
    HmiDataLogComposition dataLogs = hmiSoftware.DataLogs;
    HmiDataLog dataLogObj = dataLogs.Find(datalogName);
}
```

Arbeiten mit Datenarchiven

Mit Datenarchiven können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Datenarchive erstellen: Datalogs.Create() (Seite 703)
- Datenarchive löschen: Datalogs.Delete() (Seite 703)
- Eigenschaften von Datenarchiven: Auf Eigenschaften von Datenarchiven zugreifen (Seite 704)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Datalogs.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um ein Datenarchiv mit der Methode Create() der Klasse HmiDatalogComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void DatalogCreate()
{
    //Create Datalog
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiDatalogComposition dataLogs = hmiSoftware.DataLogs;
    HmiDatalog dataLog = dataLogs.Create("Datalog_1");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

Datalogs.Delete() (RT Unified)

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um ein Datenarchiv mit der Methode Delete() der Klasse HmiDataLog zu löschen, ändern Sie folgenden Programmcode:

```
private void DatalogDelete()
{
    //user wants to delete datalog with name "Datalog1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiDataLogComposition dataLogs = hmiSoftware.DataLogs;
    HmiDataLog dataLog = dataLogs.Find("Datalog1");
    dataLog.Delete();
}
```

Auf Eigenschaften von Datenarchiven zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in den Datenarchiven unterstützt:

Name der Eigenschaft		Datentyp	Beschreibung	Zugriff
Name		String	Gibt den Namen des Datenarchivs an.	L/S
Settings		LogSettings	Gibt die Mitglieder komplexer Eigenschaften wie StorageFolder, LogMaxSize und LogTimePeriod an, die die Einstellungen für das Archiv festlegen.	R
	StorageFolder	String	Gibt den Ordnerpfad für die Archivierung an.	L/S
	LogMaxSize	Unsigned int	Gibt die maximale Größe des Archivs auf dem Speichermedium in Megabyte an.	L/S
	LogTimePeriod	LogDuration	Gibt den vom Archiv abgedeckten maximalen Zeitraum an.	L/S
	StorageDevice	DeviceNode	Gibt das zu prüfende Speichergerät an	L/S
Segment		LogSegment	Gibt die Mitglieder komplexer Eigenschaften wie SegmentMaxSize, StartTime und SegmentTimePeriod an, die die Einstellungen für das Segment festlegen.	R
	SegmentMaxSize	Unsigned int	Gibt die maximale Größe des Archivsegments in Megabyte an.	L/S
	SegmentStartTime	DateTime	Gibt den genauen Zeitpunkt an, wenn das Archivsegment gestartet wird (zu prüfen)	L/S
	SegmentTimePeriod	SegmentDuration	Gibt den vom Archivsegment abgedeckten maximalen Zeitraum an.	L/S
Backup		LogBackup	Gibt die Mitglieder komplexer Eigenschaften wie PrimaryPath und BackupMode an, die die Einstellungen für das Archiv-Backup festlegen.	R
	BackupMode	HmiBackupMode	Gibt den Backup-Modus an.	L/S
	PrimaryPath	String	Gibt den primären Pfad für das Backup an.	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf die Eigenschaften von Datenarchiven zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void DatalogPropertiesAccess()  
{  
    //Set/Get Data Log properties.  
    HmiSoftware hmiSoftware = GetHmiSoftware();  
    HmiDataLogComposition dataLogs = hmiSoftware.DataLogs;  
    HmiDataLog dataLog = dataLogs.Find("Datalog_1");  
    string name = dataLog.Name;  
    dataLog.Settings.StorageFolder = @"C:\Logs\Log";  
    dataLog.Settings.LogMaxSize = Int32.MaxValue;  
    dataLog.Settings.LogMaxSize = 2000;  
    LogDuration duration = dataLog.Settings.LogTimePeriod;  
    double durationInDouble = duration.GetDoubleLogDuration();  
    string durationInString = duration.GetStringLogDuration();  
    duration.Days = 7;  
    duration.Hours = 23;  
    duration.Minutes = 50;  
    duration.Seconds = 0;  
    duration.Ticks = 1;  
    //Method to set log duration.  
    duration.SetLogDuration(10, 12, 4, 5, 0);  
    dataLog.Backup.BackupMode = BackupMode.NoBackup;  
    dataLog.Backup.BackupMode = BackupMode.PrimaryPath;  
    dataLog.Backup.PrimaryPath = @"C:\Logs\Backup";  
    dataLog.Segment.SegmentMaxSize = 500;  
    dataLog.Segment.SegmentStartTime = DateTime.Now;  
    dataLog.Segment.SegmentStartTime = DateTime.Now.Date;  
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

5.13.1.9 DiscreteAlarms (RT Unified)

Beschreibung DiscreteAlarms (RT Unified)

Voraussetzungen

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um an Bitmeldungen eine Aktivität auszuführen, können Sie entweder die Bitmeldungen enumerieren oder Find() verwenden.

```
private void DiscreteAlarmBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all discrete alarms of device
    HmiDiscreteAlarmComposition discreteAlarms = hmiSoftware.DiscreteAlarms;
    foreach (HmiDiscreteAlarm discreteAlarm in discreteAlarms)
    {
        //working with discrete alarms.
    }
}
```

Um anhand des Namens auf eine einzelne Bitmeldung zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SearchDiscreteAlarm()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for a specific discrete alarm from list of discrete alarms
    String discreteAlarmName = "DiscreteAlarm_1";
    HmiDiscreteAlarmComposition discreteAlarms = hmiSoftware.DiscreteAlarms;
    HmiDiscreteAlarm discreteAlarmObj = discreteAlarms.Find(discreteAlarmName);
}
```

Mit Bitmeldungen arbeiten

Sie können, während Sie TIA Portal Openness verwenden, die folgenden Aktivitäten an Variablen ausführen:

- Bitmeldungen erstellen: DiscreteAlarms.Create() (Seite 707)
- Bitmeldungen löschen: DiscreteAlarms.Delete() (Seite 707)
- Eigenschaften von Bitmeldungen: Auf Eigenschaften von Bitmeldungen zugreifen (Seite 708)
- Querverweisdienst für Bitmeldungen: Über Bitmeldungen auf den Querverweisdienst zugreifen (Seite 710)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

DiscreteAlarms.Create() (RT Unified)**Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Bitmeldung mit der Methode Create() der Klasse HmiDiscreteAlarmComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void CreateDiscreteAlarm()
{
    //Create a discrete alarm with name "Alarm1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiDiscreteAlarmComposition discreteAlarms = hmiSoftware.DiscreteAlarms;
    HmiDiscreteAlarm discreteAlarm = discreteAlarms.Create("Alarm1");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

DiscreteAlarms.Delete() (RT Unified)**Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Bitmeldung mit der Methode Delete() der Klasse HmiDiscreteAlarm class zu löschen, ändern Sie folgenden Programmcode:

```
public void DeleteDiscreteAlarm()
//User wants to delete a discrete alarm with the alarm name Alarm_1
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HDiscreteAlarmComposition discreteAlarms = hmiSoftware.DiscreteAlarms;
    HmiDiscreteAlarm discreteAlarm = discreteAlarms.Find("Alarm1");
    discreteAlarm.Delete();
}
```

Siehe auch

Projekt öffnen (Seite 131)

Verbindung zum TIA Portal aufbauen (Seite 84)

Auf Eigenschaften von Bitmeldungen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in der Bitmeldung unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
EventText	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
Id	UInt32	Gibt die ID einer Meldung an. Diese ID ist ein eindeutiger Wert für jede Bitmeldung.	L/S
AlarmClass	String	Gibt die Klasse einer Bitmeldung an.	L/S
Name	String	Gibt den Namen einer Bitmeldung an.	L/S
Priority	Byte	Gibt die Priorität für die Bitmeldung an.	L/S
Origin	String	Gibt den Ursprung einer Bitmeldung an.	L/S
Area	String	Gibt den Bereich einer Bitmeldung an.	L/S
RaisedStateTag	String	Gibt die Variable an, die die Bitmeldung auslöst.	L/S
RaisedStateTagBitNumber	UInt32	Gibt das Auslösebit an der Auslösevariable an.	L/S
TriggerMode	HmiDiscreteAlarmTriggerMode	Gibt den Auslösemodus für die Bitmeldung an.	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
InfoText	MultilingualText	Gibt den Infotext einer Bitmeldung an.	L/S
AlarmParameterTags	Object	Bei einem Array von 10 Parametern die Variablen, die jeweils den Meldeparameter angeben. Jeder Meldeparameter verwendet den Variablennamen als Zeichenkette. Diese Parameter können im Meldetext verwendet werden.	L/S
EventText1	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
EventText2	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
EventText3	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
EventText4	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
EventText5	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
EventText6	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
EventText7	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
EventText8	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
EventText9	MultilingualText	Gibt den Ereignistext von HmiDiscreteAlarm an.	L/S
AcknowledgmentStateTag	String	Gibt den Quittiervariablenzustand für die Bitmeldung an.	L/S
AcknowledgmentStateTagBitNumber	UInt32	Gibt die Bitnummer der Quittiervariablen für die Bitmeldung an.	L/S
AcknowledgmentControlTag	String	Gibt die Steuervariable für die Quittierung der Bitmeldung an.	L/S
AcknowledgmentControlTagBitNumber	UInt32	Gibt die Bitnummer der Steuervariablen für die Quittierung der Bitmeldung an.	L/S
TriggerBitAddress	String	Gibt die Adresse des Auslösebits an.	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften einer Bitmeldung zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void DiscreteAlarmPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiDiscreteAlarmComposition discreteAlarms = hmiSoftware.DiscreteAlarms;
    HmiDiscreteAlarm discreteAlarm = discreteAlarms.Find(Alarm_1);
    uint id = discreteAlarm.Id;
    discreteAlarm.Id = 10;
    int priority = discreteAlarm.Priority;
    discreteAlarm.Priority = 7;
    discreteAlarm.area = "Areal";
    string area = discreteAlarm.Area;
    discreteAlarm.Name = "NewAlarm";
    string name = discreteAlarm.Name;
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Über Bitmeldungen auf den Querverweisdienst zugreifen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um mit der Methode `GetService()` der Klasse `HmiDiscreteAlarm` auf den Querverweisdienst zuzugreifen, ändern Sie folgenden Programmcode:

```
private void getCrossReferenceServiceforDiscreteAlarm()
{
    SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
    HmiSoftware hmiSoftware = softwareContainer.Software as HmiSoftware; //Accessing lower
    levels until we reach the HmiTagComposition.
    HmiDiscreteAlarmCompositionAlarmComposition discreteAlarms = hmiSoftware.DisceteAlarms;
    HmiDiscreteAlarm discreteAlarm = discreteAlarms.Find("Alarm1");
    if (discreteAlarm != null)
    {
        try
        {
            CrossReferenceService crossReferenceService =
            discreteAlarm.GetService<CrossReferenceService>();
            .....
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.10 LoggingTags (RT Unified)**Beschreibung LoggingTags (RT Unified)****Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung `HmiSoftware` (Seite 676)

Programmcode

Um an Archivvariablen eine Aktivität auszuführen, können Sie entweder die Archivvariablen enumerieren oder Find() verwenden.

```
private void LoggingTagBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all logging tags of device
    String hmiTagName = "HMI_Tag_1";
    HmiTagComposition hmiTags = hmiSoftware.Tags;
    HmiTag hmiTag = hmiTags.Find(hmiTagName);
    HmiLoggingTagComposition loggingTags = hmiTag.LoggingTags;
    foreach (HmiLoggingTag loggingTag in loggingTags)
    {
        //Working with logging tags
    }
}
```

Um anhand des Namens auf eine Archivvariable zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SearchLoggingTag()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for specific logging tag
    String hmiTagName = "HMI_Tag_1";
    String loggingTagName = "Logging_Tag1";
    HmiTagComposition hmiTags = hmiSoftware.Tags;
    HmiTag hmiTag = hmiTags.Find(hmiTagName);
    HmiLoggingTagComposition loggingTags = hmiTag.LoggingTags;
    HmiLoggingTag loggingTagObj = loggingTags.Find(loggingTagName);
}
```

Arbeiten mit Archivvariablen

Mit Datenarchiven können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Archivvariablen erstellen: LoggingTags.Create() (Seite 713)
- Archivvariablen löschen: LoggingTags.Delete() (Seite 713)
- Eigenschaften von Archivvariablen: Auf Eigenschaften von Archivvariablen zugreifen (Seite 714)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

LoggingTags.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Archivvariable für eine Variable mit der Methode Create() der Klasse HmiLoggingTagComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void LoggingTagCreate()
{
    //Create Logging Tag for given tag.
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagComposition hmiTags = hmiSoftware.Tags;
    //Tag1 exists in TIA project.
    HmiTag hmiTag = hmiTags.Find("Tag1");
    HmiLoggingTagComposition loggingTags = hmiTag.LoggingTags;
    HmiLoggingTag loggingTag = loggingTags.Create("LoggingTag1");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

LoggingTags.Delete() (RT Unified)

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Programmcode

Um eine Archivvariable für eine Variable mit der Methode Delete() der Klasse HmiLoggingTag zu löschen, ändern Sie folgenden Programmcode:

```
private void LoggingTagDelete()
{
    //To delete a Logging tag with name "LoggingTag_1" for tag "Tag1";
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagComposition hmiTags = hmiSoftware.Tags;
    //Tag1 exists in TIA project.
    HmiTag hmiTag = hmiTags.Find("Tag1");
    HmiLoggingTagComposition loggingTags = hmiTag.LoggingTags;
    //LoggingTag1 exists in TIA project.
    HmiLoggingTag loggingTag = loggingTags.Find("LoggingTag1");
    loggingTag.Delete();
}
```

Auf Eigenschaften von Archivvariablen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in den Archivvariablen unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
AggregationDelay	System.TimeSpan	Gibt den Wert der Aggregierungsverzögerung an.	L/S
AggregationMode	HmiAggregationMode	Gibt den Wert des Aggregierungsmodus an.	L/S
Cycle	System.String	Gibt den Archivierungszyklus der Archivvariablen an.	L/S
CycleFactor	System.UInt32	Gibt den Wert des Archivierungszyklusfaktors an.	L/S
DataLog	System.String	Gibt das Datenarchiv der Archivvariablen an.	L/S
HighLimit	System.Object	Gibt den oberen Grenzwert der Archivvariablen an.	L/S
LimitScope	HmiLimitScope	Gibt die Umfangsbegrenzung der Archivvariablen an.	L/S
LoggingMode	HmiLoggingMode	Gibt den Archivierungsmodus der Archivvariablen an.	L/S
LowLimit	System.Object	Gibt den unteren Grenzwert der Archivvariablen an.	L/S
Name	System.string	Gibt den Namen der Archivvariablen an.	L/S
SmoothingDeltaValue	System.Double	Gibt den Deltawert der Archivvariablen an.	L/S
SmoothingMaxTime	System.TimeSpan	Gibt die maximale Zeitdauer der Archivvariablen an.	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
SmoothingMinTime	System.TimeSpan	Gibt die minimale Zeitdauer der Archivvariablen an.	L/S
SmoothingMode	HmiSmoothingMode	Gibt den Glättungsmodus der Archivvariablen an.	L/S
Source	System.string	Gibt die Quelle der Archivvariablen an.	L/S
TriggerMode	HmiTriggerMode	Gibt den Auslösemodus der Archivvariablen an.	L/S
TriggerTag	System.String	Gibt den Wert der Auslösevariablen an.	L/S
TriggerTagBitNumber	System.UInt32	Gibt den Wert der Bitnummer der Auslösevariablen an.	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften von Archivvariablen zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void LoggingTagPropertiesAccess()
{
    //Set and Get LoggingTag properties.
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagComposition hmiTags = hmiSoftware.Tags;
    //Tag_1 exists in TIA project.
    HmiTag hmiTag = hmiTags.Find("Tag_1");
    HmiLoggingTagComposition loggingTags = hmiTag.LoggingTags;
    //LoggingTag_1 exists in TIA project.
    HmiLoggingTag loggingTag = loggingTags.Find("LoggingTag_1");
    string name = loggingTag.Name;
    HmiSmoothingMode smoothingMode = loggingTag.SmoothingMode;
    loggingTag.SmoothingMode = HmiSmoothingMode.SwingingDoor;
    HmiLimitScope limitScope = loggingTag.LimitScope;
    loggingTag.LimitScope = HmiLimitScope.WithinLimits;
    object lowLimit = loggingTag.LowLimit;
    loggingTag.LowLimit = "-32768";
    object highLimit = loggingTag.HighLimit;
    loggingTag.HighLimit = "-3";
    TimeSpan smoothingMinTime = loggingTag.SmoothingMinTime;
    TimeSpan tsMin = TimeSpan.Parse("5:00:00");
    loggingTag.SmoothingMinTime = tsMin;
    TimeSpan smoothingMaxTime = loggingTag.SmoothingMaxTime;
    TimeSpan tsMax = TimeSpan.Parse("1000");
    loggingTag.SmoothingMaxTime = tsMax;
    double smoothingDeltaValue = loggingTag.SmoothingDeltaValue;
    loggingTag.SmoothingDeltaValue = 5;
    string dataLog = loggingTag.Name;
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.11 RuntimeSettings (RT Unified)**Beschreibung Runtime-Einstellungen (RT Unified)****Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um an einer Runtime-Einstellung eine Aktivität auszuführen, können Sie entweder die Runtime-Einstellungen enumerieren oder Find() verwenden.

```
private void RuntimeSettingsBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all runtime settings of device
    HmiRuntimeSetting runtimeSettings = hmiSoftware.RuntimeSettings;
    foreach (RuntimeSetting runtimeSetting in runtimeSettings)
    {
        //Working with runtime setting
    }
}
```

Arbeiten mit Runtime-Einstellungen

Mit Runtime-Einstellungen können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Eigenschaften von Runtime-Einstellungen: Auf Eigenschaften von Runtime-Einstellungen zugreifen (Seite 717)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Auf Eigenschaften von Runtime-Einstellungen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in Runtime-Einstellungen unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
HmiSoftware.RuntimeSettings	HmiRuntimeSetting	Gibt das Runtime-Einstellungsobjekt an.	L
RuntimeSettings.StartScreen	String	Gibt das Startbild an.	L/S
RuntimeSettings.OperateAsOpcServer	Boolean	Gibt an, dass das vorgegebene Gerät als OPC-Server fungieren soll.	L/S
RuntimeSettings.LanguageAndFonts	HmiLanguageAndFont Association	Gibt die Liste der Sprachen und Schriftarten an.	L
RuntimeSettings. LanguageAndFonts [index].Enable	Boolean	Gibt die Aktivierung/Deaktivierung der Runtime-Sprache an.	L/S
RuntimeSettings.LanguageAndFonts [index].EnableForLogging	Boolean	Gibt die Sprache für Archivierung in Runtime an.	L/S
RuntimeSettings. LanguageAndFonts [index].Order	Short	Gibt den Befehl für die Übertragung von Schriftarten zum Gerät an.	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
RuntimeSettings.LanguageAndFonts[index].Language	String	Gibt den Namen der Sprache an.	L
RuntimeSettings.LanguageAndFonts[index].FixedFont1	String	Gibt die vordefinierte Schriftartenfamilie für die Schriftartenkonfiguration an.	L
RuntimeSettings.LanguageAndFonts[index].FixedFont2	String	Gibt die vordefinierte Schriftartenfamilie für die Schriftartenkonfiguration an.	L
RuntimeSettings.LanguageAndFonts[index].FixedFont3	String	Gibt die vordefinierte Schriftartenfamilie für die Schriftartenkonfiguration an.	L
RuntimeSettings.LanguageAndFonts[index].FixedFont4	String	Gibt die vordefinierte Schriftartenfamilie für die Schriftartenkonfiguration an.	L
RuntimeSettings.GMPEnabled	Boolean	Gibt an, ob das vorgegebene Gerät für Audit Logging aktiviert oder nicht aktiviert werden soll.	L/S
RuntimeSettings.OpcUaServerRuntimeSettings	HmiOpcUaServerRuntimeSettings	Gibt die OPCUA-Einstellungen für das Gerät an.	L/S
RuntimeSettings.MaxLoginRuntimeSettings.MaxLoginErrors	UInt32	Gibt die maximale Anzahl der Anmeldeversuche an, bevor der Benutzer gesperrt wird.	L/S
RuntimeSettings.MaxLoginRuntimeSettings.EnableLockAfterNumberOfAttempts	Boolean	Aktiviert die Sperre nach Anmeldefehlern.	L/S
RuntimeSettings.ProcessDiagnosticsRuntimeSettings	HmiProcessDiagnosticsRuntimeSettings	Gibt die Runtime-Einstellung der Prozessdiagnose an.	L/S
RuntimeSettings.HmiReportingSettings	HmiReportingSettings	Gibt die Runtime-Einstellung der Berichterstellung an.	L/S
RuntimeSettings.BitSelectionStrategyForResourceLists	Enum(BitNumberEvaluationType)	Gibt die Bitauswahlstrategie für Text- und Grafiklisten an.	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

```
private void RuntimeSettingsPropertiesAccess()  
{  
    //Accessing properties of RuntimeSettings  
    HmiSoftware hmiSoftware = GetHmiSoftware();  
    HmiRuntimeSetting runtimeSettings = hmiSoftware.RuntimeSettings;  
    hmiSoftware.runtimeSettings.OpcUaServerRuntimeSettings.ActAsOPCServer = true;  
    hmiSoftware.runtimeSettings.StartScreen = "Screen_1";  
    hmiSoftware.runtimeSettings.LanguageAndFonts[0].Enable = true;  
    hmiSoftware.runtimeSettings.LanguageAndFonts[0].EnableForLogging = true;  
    hmiSoftware.runtimeSettings.GMPEnabled = true;  
    //Accessing Maximal login Runtime Settings property  
    var maxLoginRuntimeSettings = hmiSoftware.RuntimeSettings.MaxLoginRuntimeSettings;  
    maxLoginRuntimeSettings.EnableLockAfterNumberOfAttempts = true;  
    var attempts = maxLoginRuntimeSettings.EnableLockAfterNumberOfAttempts;  
    maxLoginRuntimeSettings.MaxLoginErrors = 1;  
    var maxLoginErrors = maxLoginRuntimeSettings.MaxLoginErrors;  
    //Accessing Process diagnostics Runtime Settings property  
    HmiProcessDiagnosticsRuntimeSettings hmiProcessDiagnosticsRuntimeSettings =  
    hmiSoftware.RuntimeSettings.ProcessDiagnosticsRuntimeSettings;  
    hmiProcessDiagnosticsRuntimeSettings.EnableProcessDiagnostics = true;  
    var enableProcessDiagnostics =  
    hmiProcessDiagnosticsRuntimeSettings.EnableProcessDiagnostics;  
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.12 SystemTags (RT Unified)

Beschreibung Systemvariablen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um anhand des Namens auf eine einzelne Systemvariable zuzugreifen, ändern Sie folgenden Programmcode:

```
private void AccessSystemTagName()  
{  
    HmiSoftware hmiSoftware = GetHmiSoftware();  
    // Search for specific system tag name from device  
    string systemTagName = "@SystemHealthIndex"  
    HmiSystemTagComposition hmiSystemTags = hmiSoftware.SystemTags;  
    HmiSystemTag hmiSystemTagObj = hmiSystemTags.Find(systemTagName);  
}
```

Arbeiten mit Systemvariablen

Mit Systemvariablen können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Eigenschaften von Systemvariablen: Auf Eigenschaften von Systemvariablen zugreifen (Seite 720)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Auf Eigenschaften von Systemvariablen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in Systemvariablen unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Name	String	Gibt den Namen des Systemvariablen an.	R
DataType	String	Gibt den Datentyp der Systemvariablen an.	R

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf Eigenschaften von Systemvariablen zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SystemTagPropertiesAccess()
{
    //Get the "Name" and "DataType" properties of system tag
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiSystemTagComposition hmiSystemTags = hmiSoftware.SystemTags;
    HmiSystemTag hmiSystemTag = hmiSystemTags.Find("@UserName");
    string name = hmiSystemTag.Name;
    string dataType = hmiSystemTag.DataType;
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.13 Bilder (RT Unified)**Beschreibung Bilder (RT Unified)****Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um an einem Bild eine Aktivität auszuführen, können Sie die Befehle "enumerate", "find", "index" und die Methode "Contains()" verwenden.

Um auf alle Bilder des Geräts zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ScreenBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all the screens of device
    HmiScreenComposition objScreens = hmiSoftware.Screens;
    foreach (HmiScreen ObjHmiScreen in objScreens)
    {
        //Working with Screens
    }
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um auf Bilder in Bilderlisten anhand des Namens zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ScreenSearch()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can search specific screen from screenlist
    string hmiScreenName = "HMI_Screen_1";
    HmiScreenComposition objScreens = hmiSoftware.Screens;
    HmiScreen objHmiScreen = objScreens.Find(hmiScreenName);
}
```

Um auf Bilder in Bilderlisten anhand des Index zuzugreifen, ändern Sie folgenden Programmcode:

```
private void SearchScreenByIndex()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreenComposition objScreens = hmiSoftware.Screens;
    HmiScreen objHmiScreen = objScreens[0];
}
```

Um mit der Methode Contains zu ermitteln, ob in einer Liste von Bildobjekten ein bestimmtes Bildobjekt vorhanden ist, ändern Sie folgenden Programmcode:

```
private void IsScreenExist()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreenComposition objScreens = hmiSoftware.Screens;
    HmiScreen screen1 = objScreens.Find("TestScreen1");
    bool isExist = objScreens.Contains(screen1);
}
```

Mit Bildern arbeiten

Mit Bildern können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Bilder erstellen: Screens.Create() (Seite 723)
- Bilder löschen: Screens.Delete() (Seite 723)
- Eigenschaften von Bildern: Auf Eigenschaften von Bildern zugreifen (Seite 724)
- Querverweisdienst für Bilder: Über Bilder auf den Querverweisdienst zugreifen (Seite 730)
- Eigenschaft von Bild-Ereignis-Handlern: Ereignis-Handler (Seite 731)
- Eigenschaft von Bild-Bildobjekten: Bildobjekte (Seite 734)

Siehe auch

- Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen (Seite 131)
- Beschreibung Bildobjekte (Seite 734)
- Beschreibung Ereignis-Handler (Seite 731)

Screens.Create() (RT Unified)**Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um ein Bild durch Zugreifen auf die Eigenschaft Screens des HMI Software-Objekts zu erstellen, ändern Sie folgenden Programmcode:

```
private void ScreenCreate()  
{  
    HmiSoftware hmiSoftware = GetHmiSoftware();  
    HmiScreenComposition objScreens = hmiSoftware.Screens;  
    HmiScreen objHmiScreen = objScreens.Create("TestScreen");  
}
```

Siehe auch

- Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen (Seite 131)

Screens.Delete() (RT Unified)**Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um ein Bild durch Zugreifen auf die Eigenschaft Screens des HmiSoftware-Objekts zu löschen, ändern Sie folgenden Programmcode:

```
private void DeleteScreen()
{
    //Case 1
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreenComposition objScreens = hmiSoftware.Screens;
    HmiScreen objHmiScreen = objScreens.Find("TestScreen");
    if (objHmiScreen != null)
    {
        objHmiScreen.Delete();
    }
}

private void ScreenDelete()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreenComposition objScreens = hmiSoftware.Screens;
    objScreens.Find("TestScreen1");
    IEngineeringObject alarmClassEnggObj = (IEngineeringObject)objScreens[0];
    if (alarmClassEnggObj != null)
    {
        alarmClassEnggObj.Invoke("Delete", null);
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Auf Eigenschaften von Bildern zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Bildern unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
BackColor	Color	Gibt die Hintergrundfarbe eines Bilds an.	L/S
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe eines Bilds an, die in Füllmustern verwendet wird.	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Name	String	Gibt den Namen eines Bilds an.	L/S
ScreenNumber	UInt16	Gibt die konfigurierte Nummer eines Bilds an.	L/S
BackFillPattern	HmiFillPattern	Gibt das Hintergrundfüllmuster eines Bilds an.	L/S
BackGraphic	String	Gibt die Grafik an, die im Hintergrund eines Bilds gezeigt werden soll.	L/S
HorizontalAlignment	HmiHorizontalAlignment	Gibt die aktuelle Position der Bildlaufleiste des übergeordneten HMI-Bildfensters (der obersten Ebene) an. Wenn das Bild kleiner als das übergeordnete Fenster ist, wird diese Ausrichtungsinformation zur Positionierung verwendet.	L/S
VerticalAlignment	HmiVerticalAlignment	Gibt die aktuelle Position der Bildlaufleiste des übergeordneten HMI-Bildfensters (der obersten Ebene) an. Wenn das Bild kleiner als das übergeordnete Fenster ist, wird diese Ausrichtungsinformation zur Positionierung verwendet.	L/S
BackgroundFillMode	HmiBackgroundFillMode	Gibt an, ob der Hintergrund nur das Bild oder die gesamte Fensteransicht ausfüllt.	L/S
BackGraphicStretchMode	HmiGraphicStretchMode	Gibt an, ob die Hintergrundgrafik eines Bilds gestreckt dargestellt wird.	L/S
DisplayName	String	Gibt den Anzeigenamen eines Bilds an.	L/S
Width	uint	Gibt die Abmessungen des Bilds in geräteunabhängigen Einheiten (DIU) an.	L/S
Height	uint	Gibt die Abmessungen des Bilds in geräteunabhängigen Einheiten (DIU) an.	
Enabled (AllowOperatorControl)	Boolean	Gibt an, ob das Bild funktionsfähig ist (aktivieren = "true").	L/S
Dynamizations	DynamizationBaseComposition	Gibt die Dynamisierungssammlung an.	L
Parent	IEngineeringObject	Gibt das übergeordnete Element dieses Objekts an.	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
ScreenItems	HmiScreenItem BaseComposition	Gibt die Zusammensetzung der Bildobjekte an. Weitere Informationen über Bildobjekte siehe Bildobjekte (Seite 734)	L/S
EventHandlers	HmiScreenEventHandler- Composition	Gibt die Ereignis-Handler an. Weitere Informationen über Ereignis-Handler siehe Ereignis-Handler (Seite 731)	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um die Eigenschaften eines Bilds festzulegen, ändern Sie folgenden Programmcode:

```
private void SettingPropertiesScreen()
{
    HmiSoftware hmisoftware = GetHmiSoftware();
    HmiScreenComposition objScreens = hmisoftware.Screens;
    HmiScreen objHmiScreen = objScreens.Find("TestScreen");
    objHmiScreen.Width = 50;
    objHmiScreen.ScreenNumber = 1;
    objHmiScreen.BackGraphic = "DownArrow";
    objHmiScreen.Enabled = true;
    objHmiScreen.Height = 123;
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um mit `GetAttributes ( )` alle Eigenschaften eines Bilds abzurufen, ändern Sie folgenden Programmcode:

```
private void GettingPropertiesScreen()
{
    HmiSoftware hmisoftware = GetHmiSoftware();
    HmiScreenComposition objScreens = hmisoftware.Screens;
    HmiScreen objHmiScreen = objScreens.Find("TestScreen");
    List<string> getlststring = new List<string>();
    getlststring.Add("AlternateBackColor");
    getlststring.Add("BackColor");
    getlststring.Add("BackFillPattern");
    getlststring.Add("BackGraphic");
    getlststring.Add("BackGraphicStretchMode");
    getlststring.Add("BackgroundFillMode");
    getlststring.Add("DisplayName");
    //getlststring.Add("Dynamization");
    getlststring.Add("Enabled");
    //getlststring.Add("EnabledExplicitRelease");
    getlststring.Add("Height");
    getlststring.Add("HorizontalAlignment");
    //getlststring.Add("Layers");
    getlststring.Add("Name");
    //getlststring.Add("Parent");
    //getlststring.Add("ScreenElements");
    //getlststring.Add("ScreenItems");
    getlststring.Add("ScreenNumber");
    getlststring.Add("VerticalAlignment");
    getlststring.Add("Width");
    var getallpropValue = objHmiScreen.GetAttributes(getlststring);
    foreach(var item in getallpropValue)
    {
        Console.Write(item.ToString());
        Console.Write("\n");
    }
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um mit `SetAttributes()` alle Eigenschaften eines Bilds festzulegen, ändern Sie folgenden Programmcode:

```
private void SettingPropertyValueScreen()
{
    UInt32 Height=400;
    UInt32 Width = 200;
    ushort ScreenNumber = 11;
    HmiSoftware hmisoftware = GetHmiSoftware();
    HmiScreenComposition objScreens = hmisoftware.Screens;
    HmiScreen objHmiScreen = objScreens.Find("Screen_1");
    Dictionary<string, object> setPropertyName = new Dictionary<string, object>();
    setPropertyName.Add("AlternateBackColor", Color.Aqua);
    setPropertyName.Add("BackColor", Color.Aqua);
    setPropertyName.Add("BackFillPattern", HmiFillPattern.GradientHorizontalTricolor);
    setPropertyName.Add("BackGraphic", "DownArrow");
    setPropertyName.Add("BackGraphicStretchMode", HmiGraphicStretchMode.Fill);
    setPropertyName.Add("BackgroundFillMode", HmiBackgroundFillMode.Screen);
    //setPropertyName.Add("DisplayName", "AnalogAlarmByDiffMethod");
    setPropertyName.Add("Enabled", false);
    //setPropertyName.Add("EnabledExplicitRelease", "AnalogAlarmByDiffMethod");
    setPropertyName.Add("Height", Height);
    setPropertyName.Add("HorizontalAlignment", HmiHorizontalAlignment.Right);
    setPropertyName.Add("Name", "Screen_2");
    setPropertyName.Add("ScreenNumber", ScreenNumber);
    setPropertyName.Add("VerticalAlignment", HmiVerticalAlignment.Center);
    setPropertyName.Add("Width", Width);
    objHmiScreen.SetAttributes(setPropertyName);
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um mit `IEngineeringObject`'s `GetAttribute` den Wert einer Eigenschaft abzurufen, ändern Sie folgenden Programmcode:

```
private void GettingPropertyValueScreen()
{
    HmiSoftware hmisoftware = GetHmiSoftware();
    HmiScreenComposition objScreens = hmisoftware.Screens;
    HmiScreen objHmiScreen = objScreens.Find("Testscreen");
    IEngineeringObject obj = objHmiScreen;
    Color clraltnvbk = (Color)obj.GetAttribute("AlternateBackColor");
    Color clrbk = (Color)obj.GetAttribute("BackColor");
    HmiFillPattern filptrns = (HmiFillPattern)obj.GetAttribute("BackFillPattern");
    string bkggraphic = (string)obj.GetAttribute("BackGraphic");
    HmiGraphicStretchMode strmode =
        (HmiGraphicStretchMode)obj.GetAttribute("BackGraphicStretchMode");
    HmiBackgroundFillMode bkgrndfilmode =
        (HmiBackgroundFillMode)obj.GetAttribute("BackgroundFillMode");
    MultilingualText dsplyName = (MultilingualText)obj.GetAttribute("DisplayName");
    //DynamizationBaseComposition dymztns =
        (DynamizationBaseComposition)obj.GetAttribute("Dynamizations");
    bool enable = (bool)obj.GetAttribute("Enabled");
    //SumRef objsom = (SomRef)obj.GetAttribute("EnableExplicitRelease");
    uint Height = (uint)obj.GetAttribute("Height");
    HmiHorizontalAlignment hrzntl =
        (HmiHorizontalAlignment)obj.GetAttribute("HorizontalAlignment");
    HmiVerticalAlignment vrtcle = (HmiVerticalAlignment)obj.GetAttribute("VerticalAlignment");
    //HmiLayerPartComposition objlyrs = (HmiLayerPartComposition)Obj.GetAttribute("layers");
    string name = (string)obj.GetAttribute("Name");
    //IEngineeringObject objparen = (IEngineeringObject)obj.GetAttribute("Parent");
    //HmiscreenElementBaseComposition objelemnt =
        (HmiscreenElementBaseComposition)obj.GetAttribute("ScreenElements");
    //HmiScreenItemBaseComposition objitems =
        (HmiScreenItemBaseComposition)obj.GetAttribute("ScreenItems");
    ushort scrnnumber = (ushort)obj.GetAttribute("ScreenNumber");
    uint width = (uint)obj.GetAttribute("Width");
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um festzustellen, ob alle Eigenschaften eines Bilds gültige Werte haben, ändern Sie folgenden Programmcode:

```
private void IsValidPropertyValue()
{
    HmiSoftware hmiSoftware = GetHmiSoftware ();
    HmiScreen objHmiScreen = hmiSoftware.Screens.Create("HmiScreen_1");
    IList<IValidator> objectsToValidate = new List<IValidator>();
    foreach (var item in hmiSoftware.Screens)
    {
        objectsToValidate.Add(item);
    }
    foreach (IValidator validator in objectsToValidate)
    {
        IList<HmiValidationResult> errors = validator.Validate();
        if (errors != null && errors.Count > 0)
        {
            foreach (var errornotification in errors)
            {
                var propName = errornotification.PropertyName;
                foreach (var errormessage in errornotification.Errors)
                {
                    Console.WriteLine(errormessage);
                }
            }
        }
    }
}
```

Hinweis

Wenn während des Festlegens von Eigenschaften der Wert durch den Schreibvorgang nicht festgelegt werden kann, wird eine Recoverable exception ausgelöst.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Beschreibung Bildobjekte (Seite 734)

Über Bilder auf den Querverweisdienst zugreifen (RT Unified)**Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um mit der Methode `GetService()` der Bilderklasse auf den Querverweisdienst zuzugreifen, ändern Sie folgenden Programmcode:

```
private void AccessCrossReferenceServiceforScreen()
{
    SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
    HmiSoftware hmiSoftware = softwareContainer.Software as HmiSoftware; //Accessing lower
levels until we reach the ScreenComposition.
    ScreenComposition screenComposition = hmiSoftware.Screens;
    Screen screen = screenComposition.Find("Screen_1");
    if (screen != null)
    {
        try
        {
            CrossReferenceService crossReferenceService = screen.GetService<CrossReferenceService>();
            // .....
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

Ereignis-Handler (RT Unified)**Beschreibung Ereignis-Handler (RT Unified)****Voraussetzung**

- Auf das HMI-Unified-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um auf Ereignis-Handler für Bildobjekte zuzugreifen, ändern Sie folgenden Programmcode:

```
private void AccessEvent(HmiSoftware hmiSoftware);
{
    var screenComp = ((HmiSoftware)targetSWTag).Screens;
    HmiScreen screenOne = screenComp.Create("Screen2");
    HmiScreenEventHandler screenHandler =
screenOne.EventHandlers.Create(HmiScreenEventType.Unloaded);
    HmiScreenEventHandler screenHandlerSearched =
screenOne.EventHandlers.Find(HmiScreenEventType.Unloaded);
}
```

Mit Ereignissen für Bildobjekte arbeiten

Mit Ereignissen für Bildobjekte können Sie während der Verwendung von TIA Portal Openness folgende Aktivitäten durchführen:

- Ereignis-Handler erstellen: `EventHandlers.Create()` (Seite 732)
- Ereignis-Handler löschen: `EventHandlers.Delete()` (Seite 732)
- Eigenschaften von Ereignis-Handlern: Auf Eigenschaften von Ereignis-Handlern zugreifen (Seite 733)

`EventHandlers.Create()` (RT Unified)

Voraussetzung

- Auf das HMI-Unified-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung `HMI Software` (Seite 676)

Programmcode

Um Ereignisse am Bildobjekt-Ebenenobjekt mit dem EventHandler-Navigator zu erstellen, ändern Sie folgenden Programmcode:

```
private void CreateEvents(HmiSoftware hmiSoftware)
{
    var screenComp = ((HmiSoftware)targetSWTag).Screens;
    HmiScreen screenOne = screenComp.Create(string screenName = "Screen2");
    HmiScreenEventHandler screenHandler =
screenOne.EventHandlers.Create(HmiScreenEventType.Unloaded);
}
```

`EventHandlers.Delete()` (RT Unified)

Voraussetzung

- Auf das HMI-Unified-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung `HMI Software` (Seite 676)

Programmcode

Um Ereignisse für Bilder und Bildobjekte zu löschen, ändern Sie den folgenden Programmcode:

```
private void DeletePropertyEvent(HmiSoftware hmisoftware)
{
    HmiScreen hmiscreen = hmisoftware.Screens.Create(Guid.NewGuid().ToString());
    var hmiscreenPropEventDyn = hmiscreen.EventHandlers.Create("Enabled",
PropertyEventType.Change);
    var enabledevent = hmiscreen.EventHandlers.Find("Enabled", PropertyEventType.Change);
    enabledevent.Delete();
}
```

Auf Eigenschaften von Ereignis-Handlern zugreifen (RT Unified)

Voraussetzung

- Auf das HMI-Unified-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um auf Eigenschaften von Ereignissen für Bilder und Bildobjekte zuzugreifen, ändern Sie folgenden Programmcode:

```
private void AccessEvent(HmiSoftware hmiSoftware)
{
    var screenComp = ((HmiSoftware)targetSWTag).Screens;
    HmiScreen screenOne = screenComp.Create("Screen2");
    HmiScreenEventHandler screenHandler =
screenOne.EventHandlers.Create(HmiScreenEventType.Unloaded);
    Console.WriteLine(screenOne.Name + ".EventHandler.EventType - {0}",
screenHandler.EventType);
    Console.WriteLine(screenOne.Name + ".EventHandler.Script.Async - {0}",
screenHandler.Script.Async);
    Console.WriteLine(screenOne.Name + ".EventHandler.Script.GlobalDefinitionAreaScriptCode
- {0}", screenHandler.Script.GlobalDefinitionAreaScriptCode);
    Console.WriteLine(screenOne.Name + ".EventHandler.Script.ScriptCode - {0}",
screenHandler.Script.ScriptCode);
}
```

Bildobjekte (RT Unified)

Beschreibung Bildobjekte (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um eine Aktivität an einem Bildobjekt durchzuführen, können Sie die Befehle „enumerate“, „find“, „index“ sowie die Methode „Contains()“ verwenden.

```
private void ScreenItemsBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    // User can navigate all screen items of screen
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    foreach (var item in screenitems)
    {
        //Working with screenitems
    }
}
```

Um anhand des Namens von der Bildobjekt-Liste auf ein einzelnes Bildobjekt zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SearchScreenItem()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    string hmiScreenItemName = "Line_1";
    HmiScreenItemBase screenitems = hmiScreen.ScreenItems.Find(hmiScreenItemName);
}
```

Um anhand des Indexes von der Bildobjekt-Liste auf ein einzelnes Bildobjekt zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void SearchScreenItemByIndex()
{
    HmiScreenItemBase screenitem = hmiScreen.ScreenItems[1];
}
```

Um mit der Methode `Contains()` zu ermitteln, ob in einer Liste von Bildobjekten ein bestimmtes Bildobjekt vorhanden ist, ändern Sie folgenden Programmcode:

```
private void IsScreenItemExist()  
{  
    HmiSoftware hmiSoftware = GetHmiSoftware();  
    HmiScreen hmiScreen = hmiSoftware.Screens.Create("Screen_21");  
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;  
    HmiLine hmiLine = screenitems.Create<HmiLine>("ScreenItems_1");  
    bool isexists = screenitems.Contains(hmiLine);  
}
```

Mit Bildobjekten arbeiten

Mit Bildobjekten können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Bildobjekte erstellen: `ScreenItems.Create()` (Seite 736)
- Bildobjekte löschen: `ScreenItems.Delete()` (Seite 737)
- Objektliste der Bildobjekte: Objekte der Bildobjekte (Seite 739)
- Querverweisdienst für Bildobjekte: Über Bildobjekte auf den Querverweisdienst zugreifen (Seite 837)
- Eigenschafts-Ereignis-Handler von Bildobjekten: Eigenschafts-Ereignis-Handler (Seite 835)

Siehe auch

`Screens.Create()` (Seite 723)

`Screens.Delete()` (Seite 723)

Auf Eigenschaften von Linien zugreifen (Seite 755)

Auf Eigenschaften von Polygonzügen zugreifen (Seite 757)

Auf Eigenschaften von Polygonen zugreifen (Seite 759)

Auf Eigenschaften von Ellipsen zugreifen (Seite 748)

Auf Eigenschaften von Ellipsensegmenten zugreifen (Seite 751)

Auf Eigenschaften von Kreissegmenten zugreifen (Seite 745)

Auf die Eigenschaften von Ellipsenbögen zugreifen (Seite 749)

Auf die Eigenschaften von Kreisbögen zugreifen (Seite 744)

Auf Eigenschaften von Kreisen zugreifen (Seite 742)

Auf Eigenschaften von Rechtecken zugreifen (Seite 761)

Auf Eigenschaften von Grafikanzeigen zugreifen (Seite 753)

ScreenItems.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um ein HmiLine-Bildobjekt mit Create() aus der Klasse HmiScreenItemBaseComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void ScreenItemCreate()
{
    // Creating a Hmiline screenitem
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Create("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiSoftware.Screens[0].ScreenItems;
    HmiLine hmiLine = screenitems.Create<HmiLine>("ScreenItem_1");
}
```

Um ein HmiIOField-Bildobjekt mit der Methode Create() der Klasse HmiScreenItemBaseComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void CreateScreenItem()
{
    // Creating a HmiIOField screenitem
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiIOField hmiIOField = screenitems.Create<HmiIOField>("ScreenItem_1");
}
```

Um ein HmiAlarmControl-Bildobjekt mit der Methode Create() der Klasse HmiScreenItemBaseComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void ControlScreenItemCreate()
{
    // Creating a HmiAlarmControl screenitem
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiAlarmControl hmiAlarmControl = screenitems.Create<HmiAlarmControl>("ScreenItem_12");
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um einen Bildbausteincontainer mit der Methode `create()` der Klasse `HmiScreenItemBaseComposition` zu erstellen, ändern Sie folgenden Programmcode:

```
private void createFaceplateContainer(HmiSoftware hmiSoftware, string screenName =
"screen_1",
string faceplatecontainerName = "Faceplate container_1")
{
    HmiScreen hmiScreen= hmiSoftware.Screens.Find(screenName);
    HmiScreenItemBaseComposition screenItems = hmiSoftware.ScreenItems;
    HmiFaceplateContainer faceplate =
screenItems.Create<HmiFaceplateContainer>(faceplatecontainerName);
}
```

Hinweis

Um andere Bildobjekte zu erstellen, können Sie den Bildobjekt-Typnamen angeben, der in TIA Portal Openness unterstützt wird.

ScreenItems.Delete() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung `HmiSoftware` (Seite 676)

Programmcode

Um ein `HmiLine`-Bildobjekt zu löschen, ändern Sie folgenden Programmcode:

```
private void ScreenItemsDelete()
{
    //Case 1
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiLine hmiLine = screenitems.Create<HmiLine>("ScreenItem_1");
    if (hmiLine != null)
    {
        hmiLine.Delete();
    }
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

```

private void ScreenItemDelete()
{
//Case 2
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiLine hmiLine = (HmiLine)screenitems.Find("ScreenItem_12");
    IEngineeringObject ObjhmiLineEnggObj = hmiLine;
    if (ObjhmiLineEnggObj != null)
    {
        ObjhmiLineEnggObj.Invoke("Delete", null);
    }
}

```

Um ein Bildbausteincontainer-Bildobjekt zu löschen, ändern Sie folgenden Programmcode:

```

private void DeleteFaceplateInstance(HmiSoftware hmiSoftware, string
faceplateContainerName = "Faceplatecontainer_1")
{
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiFaceplateContainer faceplate =
(HmiFaceplateContainer)hmiScreen.ScreenItems.Find(faceplateContainerName);
    If (faceplate! = null)
    {
        faceplate.Delete ();
    }
}

```

Objekte der Bildobjekte (RT Unified)

Einleitung

Die folgenden Bildobjekte werden von TIA Portal Openness unterstützt.

Namensraum/-räume	Bildobjektname	Bildobjektname	Type	Eigenschaft
Siemens.Engineering.HmiUnified.UI.Shapes Siemens.Engineering.HmiUnified.UI.Base	Stammobjekte	Circle	HmiCircle	Für Details zu den Eigenschaften eines Kreises siehe Auf Eigenschaften von Kreisen zugreifen (Seite 742)
		Circular arc	HmiCircularArc	Für Details zu den Eigenschaften eines Kreisbogens siehe Auf die Eigenschaften von Kreisbögen zugreifen (Seite 744)
		Circle Segment	HmiCircleSegment	Für Details zu den Eigenschaften eines Kreissegments siehe Auf Eigenschaften von Kreissegmenten zugreifen (Seite 745)
		Ellipse	HmiEllipse	Für Details zu den Eigenschaften einer Ellipse siehe Auf Eigenschaften von Ellipsen zugreifen (Seite 748)
		Elliptical arc	HmiEllipticalArc	Für Details zu den Eigenschaften eines Ellipsenbogens siehe Auf die Eigenschaften von Ellipsenbögen zugreifen (Seite 749)
		Ellipse Segment	HmiEllipseSegment	Für Details zu den Eigenschaften eines Ellipsensegments siehe Auf Eigenschaften von Ellipsensegmenten zugreifen (Seite 751)
		Graphic view	HmiGraphicView	Für Details zu den Eigenschaften einer Grafikanzeige siehe Auf Eigenschaften von Grafikanzeigen zugreifen (Seite 753)
		Line	HmiLine	Für Details zu den Eigenschaften einer Linie siehe Auf Eigenschaften von Linien zugreifen (Seite 755)
		Polyline	HmiPolyline	Für Details zu den Eigenschaften eines Polygonzugs, siehe Auf Eigenschaften von Polygonzügen zugreifen (Seite 757)
		Polygon	HmiPolygon	Für Details zu den Eigenschaften eines Polygons siehe Auf Eigenschaften von Polygonen zugreifen (Seite 759)
		Rectangle	HmiRectangle	Für Details zu den Eigenschaften eines Rechtecks siehe Auf Eigenschaften von Rechtecken zugreifen (Seite 761)
		Text Box	HmiTextBox	Für Details zu den Eigenschaften eines Textfelds siehe Auf Eigenschaften eines Textfelds zugreifen (Seite 763)

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Namensraum/-räume	Bildobjektname	Bildobjektname	Type	Eigenschaft
Siemens.Engineering.HmiUnified.ModernUI.Widgets Siemens.Engineering.HmiUnified.ModernUI.Base	Elemente	Bar	HmiBar	Für Details zu den Eigenschaften eines Balkens siehe Auf Eigenschaften eines Balkens zugreifen (Seite 767)
		Button	HmiButton	Für Details zu den Eigenschaften einer Schaltfläche siehe Auf Eigenschaften einer Schaltfläche zugreifen (Seite 769)
		Checkbox	HmiCheckBox-Group	Für Details zu den Eigenschaften eines Kontrollkästchens siehe Auf Eigenschaften eines Kontrollkästchens zugreifen (Seite 772)
		Clock	HmiClock	Für Details zu den Eigenschaften einer Uhr siehe Auf Eigenschaften einer Uhr zugreifen (Seite 775)
		Gauge	HmiGauge	Für Details zu den Eigenschaften eines Zeigerinstruments siehe Auf Eigenschaften eines Zeigerinstruments zugreifen (Seite 779)
		IO Field	HmiIOField	Für Details zu den Eigenschaften eines EA-Felds siehe Auf Eigenschaften eines EA-Felds zugreifen (Seite 782)
		Listbox	HmiListBox	Für Details zu den Eigenschaften eines Listenfelds siehe Auf Eigenschaften eines Listenfelds zugreifen (Seite 784)
		RadioButton	HmiRadioButton-Group	Für Details zu den Eigenschaften einer Optionsschaltfläche siehe Auf Eigenschaften einer Optionsschaltfläche zugreifen (Seite 788)
		Slider	HmiSlider	Für Details zu den Eigenschaften eines Schiebereglers siehe Auf Eigenschaften eines Schiebereglers zugreifen (Seite 791)
		Switch	HmiToggleSwitch	Für Details zu den Eigenschaften eines Schalters siehe Auf Eigenschaften eines Schalters zugreifen (Seite 795)
		Symbolic IOField	HmisymbolicIO-Field	Für Details zu den Eigenschaften eines symbolischen EA-Felds siehe Auf Eigenschaften eines symbolischen EA-Felds zugreifen (Seite 797)
		TouchArea	HmiTouchArea	Für Details zu den Eigenschaften eines Touchbereichs siehe Auf Eigenschaften eines Touchbereichs zugreifen (Seite 800)

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Namensraum/-räume	Bildobjektname	Bildobjektname	Type	Eigenschaft
Siemens.Engineering.HmiUnified.ModernUI.Controls Siemens.Engineering.HmiUnified.ModernUI.Base	Bedienelemente	Alarm Control	HmiAlarmControl	Für Details zu den Eigenschaften einer Meldeanzeige siehe Auf Eigenschaften einer Meldeanzeige zugreifen (Seite 801)
		Function Trend Control	HmiFunctionTrendControl	Für Details zu den Eigenschaften einer f(x)-Kurvenanzeige siehe Auf Eigenschaften einer f(x)-Kurvenanzeige zugreifen (Seite 803)
		Faceplate Container	HmiFaceplateContainer	Für Details zu den Eigenschaften eines Bildbaustein-Containers siehe Bildbausteincontainer (Seite 822)
		Media Player	HmiMediaControl	Für Details zu den Eigenschaften eines Media Players siehe Auf Eigenschaften eines Media Players zugreifen (Seite 806)
		Process Control	HmiProcessControl	Für Details zu den Eigenschaften einer Tabellenanzeige siehe Auf Eigenschaften einer Tabellenanzeige einstellen (Seite 808)
		Parameter Set Control	HmiDetailedParameterControl	Für Details zu den Eigenschaften einer Parametersatzanzeige siehe Auf Eigenschaften einer Parametersatzanzeige zugreifen (Seite 810)
		Screen Window	HmiScreenWindow	Für Details zu den Eigenschaften eines Bildfensters siehe Auf Eigenschaften eines Bildfensters zugreifen (Seite 813)
		System Diagnosis Control	HmiSystemDiagnosisControl	Für Details zu den Eigenschaften einer Systemdiagnoseanzeige siehe Auf Eigenschaften einer Systemdiagnoseanzeige zugreifen (Seite 814)
		Trend Control	HmiTrendControl	Für Details zu den Eigenschaften einer Kurvenanzeige siehe Auf Eigenschaften einer Kurvenanzeige zugreifen (Seite 816)
		Trend Companion	HmiTrendCompanion	Für Details zu den Eigenschaften einer Wertetabelle siehe Auf Eigenschaften einer Wertetabelle zugreifen (Seite 818)
		Web Control	HmiWebControl	Für Details zu den Eigenschaften eines Browsers siehe Auf Eigenschaften eines Browsers zugreifen (Seite 820)
	Meine Bedienelemente	Custom Web Control Container	HmiCustomWebControlContainer	Für Details zu den Eigenschaften eines Custom Web Control Containers siehe Auf Eigenschaften eines Custom Web Control Containers zugreifen (Seite 830)
	Dynamische Widgets	Custom Widget Container	HmiCustomWidgetContainer	Für Details zu den Eigenschaften eines Custom Widget Containers siehe Auf Eigenschaften eines Custom Widget Containers zugreifen (Seite 833)

Grundlegende Objekte (RT Unified)

Auf Eigenschaften von Kreisen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in Kreis-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
BorderColor	Color	Gibt die Rahmenfarbe des Kreises an.	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Kreises an.	L/S
BackColor	Color	Gibt die Hintergrundfarbe des Kreises an	L/S
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des Kreises an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Kreises an	R
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Kreises an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Kreises an.	L/S
BackFillPattern	HmiFillPattern	Gibt das Hintergrundfüllmuster des Kreises an	L/S
FillLevel	byte	Gibt den Füllstand des Kreises an	L/S
FillDirection	HmiFillDirection	Gibt die Füllrichtung des Kreises an	L/S
DashType	HmiDashType	Gibt den Linientyp des Kreises an	L/S
ShowFillLevel	bool	Gibt den Füllstand des Kreises an	L/S
Radius	uint	Gibt den Radius des Kreises an	L/S
CenterX	int	Gibt die X-Position des Kreises an	L/S
CenterY	int	Gibt die Y-Position des Kreises an	L/S
RotationAngle	short	Gibt den Drehwinkel des Kreises an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Kreises an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Kreises an	L/S
Opacity	float	Gibt die Opazität des Kreises an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen des Kreises an. Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt die Sichtbarkeit des Kreises an	L/S
TabIndex	ushort	Gibt den Tab-Index des Kreises an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Kreises an	L/S
Authorization	string	Gibt die Autorisierung des Kreises an	L/S
EventHandlers	HmiCircleEventHandlerComposition	Gibt den Ereignis-Handler des Kreises an	R

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
Dynamizations	DynamizationBaseComposition	Gibt die Dynamisierung des Kreises an	R
PropertyEventHandlers	PropertyEventHandlerComposition	Gibt den Ereignis-Handler des Kreises an	R

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf die Eigenschaften der Kreis-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void CircleScreenItemsPropertiesAccess
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiCircle circle = screenitems.Create<HmiCircle>("Default Value55647");
    //Name
    var name = circle.Name;
    circle.Name = "Default Value6834e";
    //BorderColor
    var bordercolor = circle.BorderColor;
    circle.BorderColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //BorderWidth
    var borderwidth = circle.BorderWidth;
    circle.BorderWidth = 10;
    //ToolTipText
    var tooltip = circle.ToolTipText;
    var tooltiptext = circle.ToolTipText.Items[0].Text;
    circle.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //DashType
    var dashtype = circle.DashType;
    circle.DashType = HmiDashType.Solid;
}
```

Siehe auch

ScreenItems.Create() (Seite 736)

Auf die Eigenschaften von Kreisbögen zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden in Kreisbogen-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
StartAngle	int	Gibt den Startwinkel des Kreisbogens an	L/S
AngleRange	int	Gibt den Winkelbereich des Kreisbogens an	L/S
LineColor	Color	Gibt die Linienfarbe des Kreisbogens an	L/S
AlternateLineColor	Color	Gibt die alternative Rahmenfarbe des Kreisbogens an	L/S
DashType	HmiDashType	Gibt den Linientyp des Kreisbogens an	L/S
EndType	HmiLineEndType	Gibt die Endlinie des Kreisbogens an	L/S
StartType	HmiLineEndType	Gibt die Startlinie des Kreisbogens an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Kreisbogens an	R
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Kreisbogens an	L/S
CapType	HmiCapType	Gibt den Scheitelpunkttyp des Kreisbogens an	L/S
LineWidth	byte	Gibt die Linienbreite des Kreisbogens an	L/S
Radius	uint	Gibt den Radius des Kreisbogens an	L/S
CenterX	int	Gibt die X-Position des Kreisbogens an	L/S
CenterY	int	Gibt die Y-Position des Kreisbogens an	L/S
RotationAngle	short	Gibt den Drehwinkel des Kreisbogens an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Kreisbogens an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Kreisbogens an	L/S
Opacity	float	Gibt die Opazität des Kreisbogens an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen des Kreisbogens an. Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt die Sichtbarkeit des Kreisbogens an.	L/S
TabIndex	ushort	Gibt den Tab-Index des Kreisbogens an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Kreisbogens an	L/S
Authorization	string	Gibt die Autorisierung des Kreisbogens an	L/S
EventHandlers	HmiCircularArcEventHandlerComposition	Gibt den Ereignis-Handler des Kreisbogens an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um auf die Eigenschaften der Kreisbogen-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void CircularArcScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiCircularArc circulararc = screenitems.Create<HmiCircularArc>("Default Value57");
    //Name
    var name = circulararc.Name;
    circulararc.Name = "Default Value567";
    //LineColor
    var linecolor = circulararc.LineColor;
    circulararc.LineColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //LineWidth
    var linewidth = circulararc.LineWidth;
    circulararc.LineWidth = 10;
    //ToolTipText
    var tooltip = circulararc.ToolTipText;
    var tooltiptext = circulararc.ToolTipText.Items[0].Text;
    circulararc.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //CapType
    var captype = circulararc.CapType;
    circulararc.CapType = HmiCapType.Round;
}
```

Siehe auch

ScreenItems.Create() (Seite 736)

Auf Eigenschaften von Kreissegmenten zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden in Kreissegment-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
StartAngle	int	Gibt den Startwinkel des Kreissegments an	L/S
AngleRange	int	Gibt den Winkelbereich des Kreissegments an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
BorderColor	Color	Gibt die Rahmenfarbe des Kreissegments an.	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Kreissegments an.	L/S
BackColor	Color	Gibt die Hintergrundfarbe des Kreissegments an	L/S
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des Kreissegments an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Kreissegments an	R
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Kreissegments an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Kreissegments an.	L/S
BackFillPattern	HmiFillPattern	Gibt das Hintergrundfüllmuster des Kreissegments an	L/S
FillLevel	byte	Gibt den Füllstand des Kreissegments an	L/S
FillDirection	HmiFillDirection	Gibt die Füllrichtung des Kreissegments an	L/S
DashType	HmiDashType	Gibt den Linientyp des Kreissegments an	L/S
ShowFillLevel	bool	Gibt das Anzeigeniveau des Kreissegments an	L/S
Radius	uint	Gibt den Radius des Kreissegments an	L/S
CenterX	int	Gibt die X-Position des Kreissegments an	L/S
CenterY	int	Gibt die Y-Position des Kreissegments an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Kreissegments an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Kreissegments an	L/S
RotationAngle	short	Gibt den Drehwinkel des Kreissegments an	L/S
Opacity	float	Gibt die Opazität des Kreissegments an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen des Kreissegments an. Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt die Sichtbarkeit des Kreissegments an	L/S
TabIndex	ushort	Gibt den Tab-Index des Kreissegments an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Kreissegments an	L/S
Authorization	string	Gibt die Autorisierung des Kreissegments an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
EventHandlers	HmiCircleSegmentEventHandlerComposition	Gibt den Ereignis-Handler des Kreissegments an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
Dynamizations	DynamizationBaseComposition	Gibt die Dynamisierung des Kreissegments an	R
PropertyEventHandlers	PropertyEventHandlerComposition	Gibt den Ereignis-Handler des Kreissegments an	R

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf die Eigenschaften der Kreissegment-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void CircleSegmentScreenItemPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiCircleSegment circlesegment = screenitems.Create<HmiCircleSegment>("Default Value5987");
    //Name
    var name = circlesegment.Name;
    circlesegment.Name = "Default Value59876";
    //AlternateBorderColor
    var bordercolor = circlesegment.AlternateBorderColor;
    circlesegment.AlternateBorderColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //BorderWidth
    var borderwidth = circlesegment.BorderWidth;
    circlesegment.BorderWidth = 10;
    //ToolTipText
    var tooltip = circlesegment.ToolTipText;
    var tooltiptext = circlesegment.ToolTipText.Items[0].Text;
    circlesegment.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
}
```

Siehe auch

ScreenItems.Create() (Seite 736)

Auf Eigenschaften von Ellipsen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in Ellipsen-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
BorderColor	Color	Gibt die Rahmenfarbe der Ellipse an.	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe der Ellipse an	L/S
BackColor	Color	Gibt die Hintergrundfarbe der Ellipse an	L/S
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe der Ellipse an	L/S
BorderWidth	byte	Gibt die Rahmenbreite der Ellipse an.	L/S
BackFillPattern	HmiFillPattern	Gibt das Hintergrundfüllmuster der Ellipse an	L/S
FillLevel	byte	Gibt den Füllstand der Ellipse an	L/S
FillDirection	HmiFillDirection	Gibt die Füllrichtung der Ellipse an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Ellipse an	R
RotationCenterPlace- ment	HmiRotationCenterPlace- ment	Gibt den Drehpunkt der Ellipse an	L/S
DashType	HmiDashType	Gibt den Linientyp der Ellipse an	L/S
ShowFillLevel	bool	Gibt den Füllstand der Ellipse an	L/S
RadiusX	uint	Gibt den X-Radius der Ellipse an	L/S
RadiusY	uint	Gibt den Y-Radius der Ellipse an	L/S
CenterX	int	Gibt die X-Position der Ellipse an	L/S
CenterY	int	Gibt die Y-Position der Ellipse an	L/S
RotationAngle	short	Gibt die Drehung der Ellipse an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt der Ellipse an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt der Ellipse an	L/S
Opacity	float	Gibt die Opazität der Ellipse an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen der Ellipse an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index der Ellipse an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text der Ellipse an	L/S
Authorization	string	Gibt die Autorisierung der Ellipse an	L/S
EventHandlers	HmiPolygonEventHand- lerComposition	Gibt den Ereignis-Handler der Ellipse an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf die Eigenschaften der Ellipsen-Bildobjekte zuzugreifen, ändern Sie folgenden Programmcode:

```
private void EllipseScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiEllipse ellipse = screenitems.Create<HmiEllipse>("Default Value5");
    //Name
    var name = ellipse.Name;
    ellipse.Name = "Default Value5";
    //AlternateBorderColor
    var bordercolor = ellipse.AlternateBorderColor;
    ellipse.AlternateBorderColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //BorderWidth
    var borderwidth = ellipse.BorderWidth;
    ellipse.BorderWidth = 10;
    //ToolTipText
    var tooltip = ellipse.ToolTipText;
    var tooltiptext = ellipse.ToolTipText.Items[0].Text;
    ellipse.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //BackFillPattern
    var backfill = ellipse.BackFillPattern;
    ellipse.BackFillPattern = HmiFillPattern.GradientBackwardDiagonal;
}
```

Siehe auch

ScreenItems.Create() (Seite 736)

Auf die Eigenschaften von Ellipsenbögen zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden in Ellipsenbogen-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
StartAngle	int	Gibt den Startwinkel des Ellipsenbogens an	L/S
AngleRange	int	Gibt den Winkelbereich des Ellipsenbogens an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
LineColor	Color	Gibt die Linienfarbe des Ellipsenbogens an	L/S
AlternateLineColor	Color	Gibt die alternative Rahmenfarbe des Ellipsenbogens an	L/S
StartType	HmiLineStartType	Gibt die Startlinie des Ellipsenbogens an	L/S
EndType	HmiLineEndType	Gibt die Endlinie des Ellipsenbogens an	L/S
CapType	HmiCapType	Gibt den Scheitelpunkttyp des Ellipsenbogens an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Ellipsenbogens an	R
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Ellipsenbogens an	L/S
DashType	HmiDashType	Gibt den Linientyp des Ellipsenbogens an	L/S
LineWidth	byte	Gibt die Linienbreite des Ellipsenbogens an	L/S
RadiusX	uint	Gibt den X-Radius des Ellipsenbogens an	L/S
RadiusY	uint	Gibt den Y-Radius des Ellipsenbogens an	L/S
CenterX	int	Gibt die X-Position des Ellipsenbogens an	L/S
CenterY	int	Gibt die Y-Position des Ellipsenbogens an	L/S
RotationAngle	short	Gibt den Drehwinkel des Ellipsenbogens an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Ellipsenbogens an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Ellipsenbogens an	L/S
Opacity	float	Gibt die Opazität des Ellipsenbogens an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen des Ellipsenbogens an. Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt die Sichtbarkeit des Ellipsenbogens an	L/S
TabIndex	ushort	Gibt den Tab-Index des Ellipsenbogens an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Ellipsenbogens an	L/S
Authorization	string	Gibt die Autorisierung des Ellipsenbogens an	L/S
EventHandlers	HmiEllipticalArcEventHandlerComposition	Gibt den Ereignis-Handler des Ellipsenbogens an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf die grundlegenden Eigenschaften der Ellipsenbogen-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void EllipticalArcScreenItemPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiEllipticalArc ellipticalarc = screenitems.Create<HmiEllipticalArc>("Default Value53987");
    //Name
    var name = ellipticalarc.Name;
    ellipticalarc.Name = "Default Value123124e";
    //LineColor
    var linecolor = ellipticalarc.LineColor;
    ellipticalarc.LineColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //LineWidth
    var linewidth = ellipticalarc.LineWidth;
    ellipticalarc.LineWidth = 10;
    //ToolTipText
    var tooltip = ellipticalarc.ToolTipText;
    var tooltiptext = ellipticalarc.ToolTipText.Items[0].Text;
    ellipticalarc.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //CapType
    var captype = ellipticalarc.CapType;
    ellipticalarc.CapType = HmiCapType.Round;
}
```

Siehe auch

ScreenItems.Create() (Seite 736)

Auf Eigenschaften von Ellipsensegmenten zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden in Ellipsensegment-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
StartAngle	int	Gibt den Startwinkel des Ellipsensegments an	L/S
AngleRange	int	Gibt den Winkelbereich des Ellipsensegments an	L/S
BorderColor	Color	Gibt die Rahmenfarbe des Ellipsensegments an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Ellipsensegments an	L/S
BackColor	Color	Gibt die Hintergrundfarbe des Ellipsensegments an	L/S
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des Ellipsensegments an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Ellipsensegments an.	L/S
BackFillPattern	byte	Gibt das Hintergrundfüllmuster des Ellipsensegments an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Ellipsensegments an	R
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Ellipsensegments an	L/S
FillLevel	byte	Gibt den Füllstand des Ellipsensegments an	
FillDirection	HmiFillDirection	Gibt die Füllrichtung des Ellipsensegments an	L/S
DashType	HmiDashType	Gibt den Linientyp des Ellipsensegments an	L/S
ShowFillLevel	bool	Gibt das Anzeigeniveau des Ellipsensegments an	L/S
RadiusX	uint	Gibt den X-Radius des Ellipsensegments an	L/S
RadiusY	uint	Gibt den Y-Radius des Ellipsensegments an	L/S
CenterX	int	Gibt die X-Position des Ellipsensegments an	L/S
CenterY	int	Gibt die Y-Position des Ellipsensegments an	L/S
RotationAngle	short	Gibt den Drehwinkel des Ellipsensegments an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Ellipsensegments an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Ellipsensegments an	L/S
Opacity	float	Gibt die Opazität des Ellipsensegments an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen des Ellipsensegments an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index des Ellipsensegments an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Ellipsensegments an	L/S
Authorization	string	Gibt die Autorisierung des Ellipsensegments an	L/S
EventHandlers	HmiEllipseEventHandlerComposition	Gibt den Ereignis-Handler des Ellipsensegments an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften der Ellipsensegment-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void EllipseSegmentScreenItemPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiEllipseSegment ellipsesegment = screenitems.Create<HmiEllipseSegment>("Default Value5");
    //Name
    var name = ellipse segmentsegment.Name;
    ellipse segmentsegment.Name = "Default Value56";
    //AlternateBorderColor
    var bordercolor = ellipse segmentsegment.AlternateBorderColor;
    ellipse segmentsegment.AlternateBorderColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //BorderWidth
    var borderwidth = ellipse segmentsegment.Height;
    ellipse segmentsegment.BorderWidth = 10;
    //ToolTipText
    var tooltip = ellipse segmentsegment.ToolTipText;
    var tooltiptext = ellipse segmentsegment.ToolTipText.Items[0].Text;
    ellipse segmentsegment.ToolTipText.Items[0].Text =
    "<body><p>TestforMultilingualProperty</p></body>";
    //BackFillPattern
    var backfill = ellipse segmentsegment.BackFillPattern;
    ellipse segmentsegment.BackFillPattern = HmiFillPattern.GradientBackwardDiagonal;
}
```

Auf Eigenschaften von Grafikanzeigen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in den Grafikansicht-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
GraphicStretch-Mode	HmiGraphicStretchMode	Gibt die Skalierung der Hintergrundgrafik an	L/S
FillLevel	byte	Gibt den Füllstand der Grafikanzeige an	L/S
FillDirection	HmiFillDirection	Gibt die Füllrichtung der Grafikanzeige an	False
ShowFillLevel	bool	Gibt die Anzeigeebene der Grafikanzeige an	False
BackFillPattern	HmiFillPattern	Gibt das Hintergrundfüllmuster der Grafikanzeige an	L/S
AlternateBack-Color	Color	Gibt die alternative Hintergrundfarbe der Grafikanzeige an	L/S
Enabled	Bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Grafikanzeige an	R

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
RotationCenter-Placement	HmiRotationCenterPlacement	Gibt den Drehpunkt der Grafikanzeige an	L/S
Top	int	Gibt die Y-Position der Grafikanzeige an	L/S
Left	int	Gibt die X-Position der Grafikanzeige an	L/S
Width	uint	Gibt die Breite der Grafikanzeige an	L/S
Height	uint	Gibt die Höhe der Grafikanzeige an	L/S
Padding	HmiPaddingPart	Gibt den Abstand der Grafikanzeige an	R
RotationAngle	short	Gibt den Drehwinkel der Grafikanzeige an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt der Grafikanzeige an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt der Grafikanzeige an	L/S
Opacity	float	Gibt die Opazität der Grafikanzeige an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen der Grafikanzeige an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt die Sichtbarkeit der Grafikanzeige an	L/S
Graphic	string	Gibt die Grafik der Grafikanzeige an	L/S
TabIndex	ushort	Gibt den Tab-Index der Grafikanzeige an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text der Grafikanzeige an	L/S
Authorization	string	Gibt die Autorisierung der Grafikanzeige an	L/S
EventHandlers	HmiGraphicViewEventHandlerComposition	Gibt den Ereignis-Handler der Grafikanzeige an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften der Grafikanzeige-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void GraphicViewScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiGraphicView graphicview = screenitems.Create<HmiGraphicView>("Default Value35487");
    //Name
    var name = graphicview.Name;
    graphicview.Name = "Default Value765";
    //AlternateBackColor
    var backcolor = graphicview.AlternateBackColor;
    graphicview.AlternateBackColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //Height
    var height = graphicview.height;
    graphicview.Height = 10;
    //ToolTipText
    var tooltip = graphicview.ToolTipText;
    var tooltiptext = graphicview.ToolTipText.Items[0].Text;
    graphicview.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //Padding
    var padding = graphicview.Padding;
    var bottom = padding.Bottom;
    padding.Bottom = 50;
    var left = padding.Left;
    padding.Left = 60;
    var right = padding.Right;
    padding.Right = 70;
    var top = padding.Top;
    padding.Top = 50;
}
```

Siehe auch

ScreenItems.Create() (Seite 736)

Auf Eigenschaften von Linien zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in einem Linien-Bildobjekt unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
X1	int	Gibt den Startpunkt der X-Achse an	L/S
Y1	int	Gibt den Startpunkt der Y-Achse an	L/S
X2	int	Gibt den Endpunkt der X-Achse an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
Y2	int	Gibt den Endpunkt der Y-Achse an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Linie an	R
LineColor	Color	Gibt die Farbe der Linie an.	L/S
AlternateLineColor	Color	Gibt die alternative Farbe der Linie an	L/S
DashType	HmiDashType	Gibt den Linientyp an	L/S
EndType	HmiLineEndType	Gibt das Ende der Linie an	L/S
StartType	HmiLineStartType	Gibt den Beginn der Linie an	L/S
CapType	HmiCapType	Gibt den Scheitelpunkttyp an	L/S
LineWidth	byte	Gibt die Breite der Linie an	L/S
Top	int	Gibt die Y-Position der Linie an	L/S
Left	int	Gibt die X-Position der Linie an	L/S
Width	uint	Gibt die Breite der Linie an	L/S
Height	uint	Gibt die Höhe der Linie an	False
RotationAngle	short	Gibt die Drehung der Linie an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt der Linie an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt der Linie an	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt der Linie an	L/S
Opacity	float	Gibt die Opazität der Linie an Die Opazität der Linie sollte zwischen 0 und 1 liegen	L/S
Name	string	Gibt den Namen der Linie an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt die Sichtbarkeit der Linie an	L/S
TabIndex	ushort	Gibt den Tab-Index der Linie an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip der Linie an	L/S
Authorization	string	Gibt die Autorisierung der Linie an	L/S
EventHandlers	HmiLineEventHandlerComposition	Gibt den Ereignis-Handler der Linie an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften der Linien-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void LineScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiLine hmiline = (HmiLine)screenitems.Find("Line_1");
    //Name
    var name = hmiline.Name;
    hmiline.Name = "Default Value";
    //AlternateLineColor
    var linecolor = hmiline.AlternateLineColor;
    hmiline.AlternateLineColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //Height
    var height = hmiline.Height;
    hmiline.Height = 100;
    //ToolTipText
    var tooltip = hmiline.ToolTipText;
    var tooltiptext = hmiline.ToolTipText.Items[0].Text;
    hmiline.ToolTipText.Items[0].Text = "<body><p>TestforMultilinugualProperty</p></body>";
    //DashType
    var dashtype = hmiline.DashType;
    hmiline.DashType = HmiDashType.DashDotDot;
}
```

Siehe auch

ScreenItems.Create() (Seite 736)

Auf Eigenschaften von Polygonzügen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in einem Polygonzug-Bildobjekt unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
LineColor	Color	Gibt die Linienfarbe des Polygonzugs an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Polygonzugs an	R
RotationCenterPlace- ment	HmiRotationCenterPlace- ment	Gibt den Drehpunkt des Polygonzugs an	L/S
AlternateLineColor	Color	Gibt die alternative Farbe des Polygonzugs an	R

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
DashType	HmiDashType	Gibt den Linientyp des Polygonzugs an	R
EndType	HmiLineEndType	Gibt den Endtyp des Polygonzugs an	R
StartType	HmiLineStartType	Gibt den Beginn der Polygonzug-Linie an	R
CapType	HmiCapType	Gibt den Scheitelpunkttyp des Polygonzugs an	R
LineWidth	byte	Gibt die Linienbreite des Polygonzugs an	R
JoinType	HmiLineJoinType	Gibt den Verknüpfungstyp im Polygonzug an	R
Top	int	Gibt die Y-Position des Polygonzugs an	R
Left	int	Gibt die X-Position des Polygonzugs an	R
Width	uint	Gibt die Breite des Polygonzugs an	R
Height	uint	Gibt die Höhe des Polygonzugs an	R
RotationAngle	short	Gibt die Drehung des Polygonzugs an	R
RotationCenterX	float	Gibt den X-Drehpunkt des Polygonzugs an	R
RotationCenterY	float	Gibt den Y-Drehpunkt des Polygonzugs an	R
Opacity	float	Gibt die Opazität des Polygonzugs an Der Wert der Opazität sollte zwischen 0 - 1 liegen	R
Name	string	Gibt den Namen des Polygonzugs an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	R
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index des Polygonzugs an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Polygonzugs an	L/S
Authorization	string	Gibt die Autorisierung des Polygonzugs an	L/S
EventHandlers	HmiPolylineEventHandlerComposition	Gibt den Ereignis-Handler des Polygonzugs an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften der Polygonzug-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void PolylineScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiPolyline polyline = (HmiPolyline)screenitems.Find("Default Value123");
    //Name
    var name = polyline.Name;
    polyline.Name = "Default Value123";
    //AlternateLineColor
    var linecolor = polyline.AlternateLineColor;
    polyline.AlternateLineColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //Height
    var height = polyline.Height;
    polyline.Height = 100;
    //ToolTipText
    var tooltip = polyline.ToolTipText;
    var tooltiptext = polyline.ToolTipText.Items[0].Text;
    polyline.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //Joint Type
    var dashtype = polyline.JoinType;
    polyline.JoinType = HmiLineJoinType.Miter;
    //Points
    var points = polyline.Points;
    var point = points[0];
    var x = point.X;
    point.X = 10;
    var y = point.Y;
    point.Y = 10;
    var newPoint = points.Create(15, 100);
}
```

Siehe auch

[ScreenItems.Create\(\)](#) (Seite 736)

Auf Eigenschaften von Polygonen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in Polygon-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
BorderColor	Color	Gibt die Rahmenfarbe des Polygons an.	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Polygons an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
BackColor	Color	Gibt die Hintergrundfarbe des Polygons an.	L/S
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des Polygons an.	L/S
Points	HmiPointComposition	Gibt den Punkt des Polygons an	R
BorderWidth	byte	Gibt die Rahmenbreite des Polygons an.	L/S
BackFillPattern	HmiFillPattern	Gibt das Hintergrundfüllmuster des Polygons an	L/S
FillLevel	byte	Gibt den Füllstand des Polygons an	L/S
FillDirection	HmiFillDirection	Gibt die Füllrichtung des Polygons an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Polygons an	R
RotationCenterPlace- ment	HmiRotationCenterPlace- ment	Gibt den Drehpunkt des Polygons an	L/S
DashType	HmiDashType	Gibt den Linientyp des Polygons an	L/S
ShowFillLevel	bool	Gibt den Füllstand des Polygons an	L/S
JoinType	HmiLineJoinType	Gibt den Verknüpfungstyp im Polygon an	L/S
Top	int	Gibt die Y-Position des Polygons an	L/S
Left	int	Gibt die X-Position des Polygons an	L/S
Width	uint	Gibt die Breite des Polygons an	L/S
Height	uint	Gibt die Höhe des Polygons an	L/S
RotationAngle	short	Gibt die Drehung des Polygons an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Polygons an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Polygons an	L/S
Opacity	float	Gibt die Opazität des Polygons an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen des Polygons an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index des Polygons an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Polygons an	L/S
Authorization	string	Gibt die Autorisierung des Polygons an	L/S
EventHandlers	HmiPolygonEventHand- lerComposition	Gibt den Ereignis-Handler des Polygons an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften der Polygon-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void PolygonScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiPolygon polygon = (HmiPolygon)screenitems.Find("Default Value563");
    //Name
    var name = polygon.Name;
    polygon.Name = "Default Value563";
    //AlternateBorderColor
    var linecolor = polygon.AlternateBorderColor;
    polygon.AlternateBorderColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //Height
    var height = polygon.Height;
    polygon.Height = 100;
    //ToolTipText
    var tooltip = polygon.ToolTipText;
    var tooltiptext = polygon.ToolTipText.Items[0].Text;
    polygon.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //BackFillPattern
    var backfill = polygon.BackFillPattern;
    polygon.BackFillPattern = HmiFillPattern.GradientBackwardDiagonal;
    //Points
    var points = polygon.Points;
    var point = points[0];
    var x = point.X;
    point.X = 10;
    var y = point.Y;
    point.Y = 10;
    var newPoint = points.Create(15, 100);
}
```

Siehe auch

Screens.Create() (Seite 723)

Auf Eigenschaften von Rechtecken zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in Rechteck-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
BorderColor	Color	Gibt die Rahmenfarbe des Rechtecks an.	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Rechtecks an.	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
BackColor	Color	Gibt die Hintergrundfarbe des Rechtecks an.	L/S
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des Rechtecks an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Rechtecks an.	L/S
BackFillPattern	HmiFillPattern	Gibt das Hintergrundfüllmuster des Rechtecks an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	False
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Rechtecks an	True
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Rechtecks an	False
Corners	HmiCornersPart	Gibt die Ecke des Rechtecks an	True
FillLevel	byte	Gibt den Füllstand des Rechtecks an	False
FillDirection	HmiFillDirection	Gibt die Füllrichtung des Rechtecks an	False
DashType	HmiDashType	Gibt den Linientyp des Rechtecks an	False
ShowFillLevel	bool	Gibt die Anzeigenebene des Rechtecks an	False
Top	int	Gibt die Y-Position des Rechtecks an	False
Left	int	Gibt die X-Position des Rechtecks an	False
Width	uint	Gibt die Breite des Rechtecks an	False
Height	uint	Gibt die Höhe des Rechtecks an	False
RotationAngle	short	Gibt den Drehwinkel des Rechtecks an	False
RotationCenterX	float	Gibt den X-Drehpunkt des Rechtecks an	False
RotationCenterY	float	Gibt den Y-Drehpunkt des Rechtecks an	False
Opacity	float	Gibt die Opazität des Rechtecks an Der Wert der Opazität sollte zwischen 0 - 1 liegen	False
Name	string	Gibt den Namen des Rechtecks an. Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	False
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	False
TabIndex	ushort	Gibt den Tab-Index des Rechtecks an	False
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Rechtecks an	False
Authorization	string	Gibt die Autorisierung des Rechtecks an	False
EventHandlers	HmiRectangleEventHandlerComposition	Gibt den Ereignis-Handler des Rechtecks an	True
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	False

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften der Rechteck-Bildobjekte zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void RectangleScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiRectangle rectangle = screenitems.Create<HmiRectangle>("Default Value3587");
    //Name
    var name = rectangle.Name;
    rectangle.Name = "Default Value57";
    //BorderColor
    var bordercolor = rectangle.BorderColor;
    rectangle.BorderColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //BorderWidth
    var borderwidth = rectangle.BorderWidth;
    rectangle.BorderWidth = 10;
    //ToolTipText
    var tooltip = rectangle.ToolTipText;
    var tooltiptext = rectangle.ToolTipText.Items[0].Text;
    rectangle.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //Corners
    var corner = rectangle.Corners;
    var bottomleftradius = corner.BottomLeftRadius;
    corner.BottomLeftRadius = 50;
    var bottomrightradius = corner.BottomRightRadius;
    corner.BottomRightRadius = 30;
    var topleftradius = corner.TopLeftRadius;
    corner.TopLeftRadius = 50;
    var toprightradius = corner.TopRightRadius;
    corner.TopRightRadius = 30;
}
```

Siehe auch

ScreenItems.Create() (Seite 736)

Auf Eigenschaften eines Textfelds zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Textfeld-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ReadOnly	bool	Gibt den Nur-Lesen-Zugriff auf das Textfeld an	L/S
TextWrapping	HmiTextWrapping	Gibt den Textumbruch des Textfelds an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
VerticalTextAlign-ment	HmiVerticalAlignment	Gibt die vertikale Textausrichtung des Textfelds an	L/S
HorizontalTextAlign-ment	HmiHorizontalAlignment	Gibt die horizontale Textausrichtung des Textfelds an	L/S
TextTrimming	HmiTextTrimming	Gibt die Textkürzung des Textfelds an	L/S
ForeColor	Color	Gibt die Vordergrundfarbe des Textfelds an	L/S
BorderColor	Color	Gibt die Rahmenfarbe des Textfelds an	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Textfelds an	L/S
Authorization	string	Gibt die Autorisierung des Textfelds an	L/S
BackColor	Color	Gibt die Hintergrundfarbe des Textfelds an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Textfelds an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Textfelds an	L
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
Font	HmiFontPart	Gibt die Schriftart des Textfelds an	L/S
Height	uint	Gibt die Höhe des Textfelds an	L/S
Top	int	Gibt die X-Position des Textfelds an	L/S
Left	int	Gibt die Y-Position des Textfelds an	L/S
Name	string	Gibt den Namen des Textfelds an	L/S
RotationAngle	short	Gibt den Verbindungsstatus des Textfelds an	L/S
VisualizeQuality	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des Textfelds an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Textfelds an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Textfelds an	L/S
Opacity	float	Gibt die Opazität des Textfelds an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
RotationCenterPlace-ment	HmiRotationCenterPlace-ment	Gibt den Drehwinkel des Textfelds an	L/S
Visible	bool	Gibt die Sichtbarkeit des Textfelds an	L/S
Width	uint	Gibt die Breite des Textfelds an	L/S
TabIndex	ushort	Gibt den Tab-Index des Textfelds an	L/S
ToolTipText	string	Gibt den Tooltip-Text des Textfelds an	L/S
EventHandlers	HmiTextBoxEventHandler Composition	Gibt den Ereignis-Handler des Textfelds an	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
Text	MultilingualText	Gibt den Text des Textfelds an	L/S
Padding	HmiPaddingPart	Gibt den Innenabstand (Padding) des Textfelds an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften eines Textfelds zuzugreifen, ändern Sie folgenden Programmcode:

```
private void TextboxPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiTextBox textbox = screenitems.Create<HmiTextBox>("Default Value334554");
    // To access basic property of textbox
    //Name
    var name = textbox.Name;
    textbox.Name = "Default Value";
    //BorderColor
    var bordercolor = textbox.BorderColor;
    textbox.BorderColor = Color.FromArgb(0xCC, 0xF0, 0x80, 0x80);
    //BorderWidth
    var borderwidth = textbox.BorderWidth;
    textbox.BorderWidth = 10;
    // To access multilingualproperty
    //ToolTipText
    var tooltip = textbox.ToolTipText;
    var tooltiptext = textbox.ToolTipText.Items[0].Text;
    textbox.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //To access other typical property of textbox
    //Font
    var font = textbox.Font;
    var italic = font.Italic;
    font.Italic = false;
    var fontname = font.FontName;
    font.FontName = HmiFontName.SimSun;
    var size = font.Size;
    font.Size = 10.2f;
    var stike = font.StrikeOut;
    font.StrikeOut = true;
    var.underline = font.Underline;
    font.Underline = false;
    font.Bold = true;
}
```

Elemente (RT Unified)

Auf Eigenschaften eines Balkens zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Balken-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des Balkens an	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Balkens an	L/S
Authorization	string	Gibt die Autorisierung des Balkens an	L
BackColor	Color	Gibt die Hintergrundfarbe des Balkens an	L/S
BorderColor	Color	Gibt die Rahmenfarbe des Balkens an	L/S
BarMode	HmiBarMode		L/S
BorderWidth	byte	Gibt die Rahmenbreite des Balkens an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Balkens an	L
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
Font	HmiFontPart	Gibt die Schriftart des Balkens an	L/S
Label	HmiTextPart	Gibt die Beschriftung des Balkens an	L/S
Height	uint	Gibt die Höhe des Balkens an	L/S
StraightScale	HmiStraightScalePart	Gibt die lineare Skala des Balkens an	L/S
Top	int	Gibt die X-Position des Balkens an	L/S
Left	int	Gibt die Y-Position des Balkens an	L/S
Name	string	Gibt den Namen des Balkens an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
NormalRangeColor	Color	Gibt die Farbe des Normalbereichs des Balkens an	L/S
OriginValue	double	Gibt den Ursprungswert des Balkens an	L/S
OutputFormat	string	Gibt das Ausgabeformat des Balkens an Der Wert des Ausgabeformats sollte zwischen 1 - 128 liegen	L/S
PeakIndicators	HmiPeakIndicator	Gibt die Spitzenwert-Anzeige des Balkens an	L/S
Process Value	string	Gibt den Prozesswert des Balkens an	L/S
ProcessValueIndicator-BackColor	Color	Gibt die Hintergrundfarbe der Prozesswertanzeige des Balkens an	L/S
ProcessValueIndicatorForeColor	Color	Gibt die Vordergrundfarbe der Prozesswertanzeige des Balkens an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
ProcessValueIndicator-Mode	HmiProcessIndicatorMode	Gibt den Modus der Prozesswertanzeige des Balkens an	L/S
RelativeToOrigin	bool	Gibt die Prozentberechnung des Ursprungswerts des Balkens an	L/S
RotationAngle	short	Gibt den Drehwinkel des Balkens an	L/S
VisualizeQuality	bool	Gibt die Verbindungsqualität des Balkens an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Balkens an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Balkens an	L/S
Opacity	float	Gibt die Opazität des Balkens an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Balkens an	L/S
ScaleBackColor	uint16	Gibt die Hintergrundfarbe der Skala des Balkens an	L/S
ScaleForeColor	Color	Gibt die Vordergrundfarbe der Skala des Balkens an	L/S
ShowTrendIndicator	bool	Gibt die Kurvenanzeige des Balkens an	L/S
Visible	bool	Gibt die Sichtbarkeit des Balkens an	L/S
Width	uint	Gibt die Breite des Balkens an	L/S
TabIndex	ushort	Gibt den Tab-Index des Balkens an	L/S
TrendIndicateColor	Color	Gibt die Farbe der Kurvenanzeige des Balkens an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Balkens an	L/S
EventHandlers	HmiBarEventHandlerComposition	Gibt den Ereignis-Handler des Balkens an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
Title	HmiTextPart	Gibt den Titel des Balkens an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften von Balken-Bildobjekten zuzugreifen, ändern Sie folgenden Programmcode:

```
private void BarScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiBar bar = screenitems.Create<HmiBar>("Default Value334554");
    //Name
    var name = bar.Name;
    bar.Name = "Default Value";
    //RotationCenterPlacement
    var rotation = bar.RotationCenterPlacement;
    bar.RotationCenterPlacement = HmiRotationCenterPlacement.AbsoluteToContainer;
    //Width
    var width = bar.Width;
    bar.Width = 100;
    //ToolTipText
    var tooltip = bar.ToolTipText;
    var tooltiptext = bar.ToolTipText.Items[0].Text;
    bar.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //Font
    var title = bar.Title;
    title.Visible = false;
    title.ForeColor = Color.Black;
    var font = title.Font;
    var italic = font.Italic;
    font.Italic = false;
    var fontname = font.Name;
    font.Name = HmiFontName.SimSun;
    var size = font.Size;
    font.Size = 10.2f;
    var stike = font.StrikeOut;
    font.StrikeOut = true;
    var underline = font.Underline;
    font.Underline = false;
    font.Weight = HmiFontWeight.Bold;
}
```

Auf Eigenschaften einer Schaltfläche zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Schaltflächen-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
AlternateBackgroundColor	Color	Gibt die alternative Hintergrundfarbe der Schaltfläche an	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe der Schaltfläche an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
BackgroundColor	Color	Gibt die Hintergrundfarbe der Schaltfläche an	L/S
BorderColor	Color	Gibt die Rahmenfarbe der Schaltfläche an	L/S
BorderWidth	byte	Gibt die Rahmenbreite der Schaltfläche an	L/S
Contents	HmiContentPart	Gibt den Inhalt der Schaltfläche an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Schaltfläche an	L
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
Font	HmiFontPart	Gibt die Schriftart der Schaltfläche an	L/S
ForegroundColor	Color	Gibt die Vordergrundfarbe der Schaltfläche an	L/S
Graphic	string	Gibt die Grafik der Schaltfläche an Der Wert der Grafik sollte zwischen 1 - 128 liegen	L/S
Height	uint	Gibt die Höhe der Schaltfläche an	L/S
Top	int	Gibt die Y-Position der Schaltfläche an	L/S
Left	int	Gibt die X-Position der Schaltfläche an	L/S
Rotation	short	Gibt die Drehung der Schaltfläche an	L/S
VisualizeQuality	bool	Gibt die visuelle Qualität der Schaltfläche an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt der Schaltfläche an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt der Schaltfläche an	L/S
Style Item Appearance	HmiStyleItem Appearance	Gibt den Stil einer Schaltfläche an	L/S
Opacity	float	Gibt die Opazität der Schaltfläche an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen der Schaltfläche an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Padding	HmiPaddingPart	Gibt den Innenabstand (Padding) der Schaltfläche an	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt der Schaltfläche an	L/S
TabIndex	ushort	Gibt den Tab-Index der Schaltfläche an	L/S
Text	MultiLingualText	Gibt den Text der Schaltfläche an	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ToolTipText	MultilingualText	Gibt den Tooltip-Text der Schaltfläche an	L
Visible	bool	Gibt die Sichtbarkeit der Schaltfläche an	L/S
Width	uint	Gibt die Breite der Schaltfläche an	L/S
EventHandlers	buttonEventHandler-Composition	Gibt den Ereignis-Handler der Schaltfläche an	L
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
AlternateGraphic	string	Gibt die Grafik der gedrückten Schaltfläche an Der Wert der alternativen Grafik sollte zwischen 1 - 128 liegen	L/S
AlternateText	MultiLingualText	Gibt den Text einer gedrückten Schaltfläche an	False

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften von Schaltflächen-Bildobjekten zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ButtonScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiButton button = screenitems.Create<HmiButton>("Default Value354");
    //Name
    var name = button.Name;
    button.Name = "Default Value";
    //AlternateBackColor
    var altbackcolor = button.AlternateBackColor;
    button.AlternateBackColor = Color.Beige;
    //Height
    var height = button.Height;
    button.Height = 100;
    //ToolTipText
    string culture = "en-US";
    Language lang = Tiaproject.LanguageSettings.Languages.Find(new CultureInfo(culture));
    MultilingualText mltprop = button.ToolTipText;
    MultilingualTextItemComposition textItemComp = mltprop.Items;
    MultilingualTextItem mlttextitem = textItemComp.Find(lang);
    mlttextitem.Text = <body><p>"TestforMultilingualProperty"</p></body>;
    //Padding
    var padding = button.Padding;
    var bottom = padding.Bottom;
    padding.Bottom = 50;
    padding.Left = 60;
    padding.Right = 70;
    padding.Top = 50;
}
```

Auf Eigenschaften eines Kontrollkästchens zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Kontrollkästchen-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
AlternateBackgroundColor	Color	Gibt die alternative Hintergrundfarbe des Kontrollkästchens an	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Kontrollkästchens an	L/S
Authorization	string	Gibt die Autorisierung des Kontrollkästchens an	L/S
BackgroundColor	Color	Gibt die Hintergrundfarbe des Kontrollkästchens an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
BorderColor	Color	Gibt die Rahmenfarbe des Kontrollkästchens an	L
BorderWidth	byte	Gibt die Rahmenbreite des Kontrollkästchens an	L/S
Content	HmiContentPart	Gibt den Inhalt des Kontrollkästchens an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Kontrollkästchens an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
Font	HmiFontPart	Gibt die Schriftart des Kontrollkästchens an	L
ForegroundColor	Color	Gibt die Vordergrundfarbe des Kontrollkästchens an	L/S
Height	uint	Gibt die Höhe des Kontrollkästchens an	L/S
Top	int	Gibt die Y-Position des Kontrollkästchens an	L/S
Left	int	Gibt die X-Position des Kontrollkästchens an	L/S
Name	string	Gibt den Namen des Kontrollkästchens an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Rotation	short	Gibt die Drehung des Kontrollkästchens an	L/S
VisualizeQuality	bool	Gibt die Verbindungsqualität des Kontrollkästchens an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Kontrollkästchens an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Kontrollkästchens an	L/S
Opacity	float	Gibt die Opazität des Kontrollkästchens an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Process value	string	Gibt den Prozesswert des Kontrollkästchens an	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Kontrollkästchens an	L/S
SelectedItemHeight	uint16	Gibt die ausgewählte Höhe des Kontrollkästchens an	L/S
SelectionItems	HmiSelectedItemPartComposition	Gibt die ausgewählten Objekte des Kontrollkästchens an	L
TabIndex	ushort	Gibt den Tab-Index des Kontrollkästchens an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Kontrollkästchens an	L
Visible	bool	Gibt die Sichtbarkeit des Kontrollkästchens an	L/S
Width	uint	Gibt die Breite des Kontrollkästchens an	L/S
EventHandlers	HmiCheckBoxGroupEventHandler Composition	Gibt den Ereignis-Handler des Kontrollkästchens an	L
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
Padding	HmiPaddingPart	Gibt den Innenabstand (Padding) des Kontrollkästchens an	L/S
SelectionPosition	HmiHorizontalAlignment	Gibt die Auswahlposition des Kontrollkästchens an	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften von Kontrollkästchen-Bildobjekten zuzugreifen, ändern Sie folgenden Programmcode:

```
private void CheckboxScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiCheckbox chkbox = screenitems.Create<HmiCheckbox>("Default Value1354");
    //Name
    var name = chkbox.Name;
    chkbox.Name = "DefaultName";
    //AlternateBackColor
    var altbackcolor = chkbox.AlternateBackColor;
    chkbox.AlternateBackColor = Color.Beige;
    //Height
    var height = chkbox.Height;
    chkbox.Height = 100;
    //ToolTipText
    var tooltip = chkbox.ToolTipText;
    var tooltiptext = chkbox.ToolTipText.Items[0].Text;
    chkbox.ToolTipText.Items[0].Text = "TestforMultilingualProperty";
    //Font
    var font = chkbox.Font;
    var italic = font.Italic;
    font.Italic = false;
    var fontname = font.Name;
    font.Name = HmiFontName.SimSun;
    var size = font.Size;
    font.Size = 10.2f;
    var stike = font.StrikeOut;
    font.StrikeOut = true;
    var underline = font.Underline;
    font.Underline = false;
    font.Weight = HmiFontWeight.Bold;
}
```

Auf Eigenschaften einer Uhr zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Uhr-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
AlternativeBackground-Color	Color	Gibt die alternative Hintergrundfarbe der Uhr an	L/S
AlternativeBorderColor	Color	Gibt die alternative Rahmenfarbe der Uhr an	L/S
BorderColor	Color	Gibt die Rahmenfarbe der Uhr an	L/S
Authorization	string	Gibt die Autorisierung der Uhr an	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
BackgroundColor	Color	Gibt die Hintergrundfarbe der Uhr an	L/S
BorderWidth	byte	Gibt die Rahmenbreite der Uhr an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Uhr an	True
DialBackColor	Color	Gibt die Hintergrundfarbe des Zifferblatts der Uhr an	False
DialLabelColor	Color	Gibt die Farbe von Beschriftungen auf dem Zifferblatt der Uhr an	False
DialMode	HmiScaleMode	Gibt den Zifferblatt-Modus der Uhr an	False
DialTickColor	Color	Gibt die Farbe von Teilstrichen auf dem Zifferblatt der Uhr an	False
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	False
Height	uint	Gibt die Höhe der Uhr an	False
Top	int	Gibt die Y-Position der Uhr an	False
Left	int	Gibt die X-Position der Uhr an	False
Name	string	Gibt den Namen der Uhr an	False
VisualizeQuality	bool	Gibt die Visualisierungsqualität der Uhr an	False
RotationCenterX	float	Gibt den X-Drehpunkt der Uhr an	False
RotationCenterY	float	Gibt den Y-Drehpunkt der Uhr an	False
ShowHours	bool	Gibt an, ob die Anzeige von Stunden aktiviert ist	False
ShowMinutes	bool	Gibt an, ob die Anzeige von Minuten aktiviert ist	False
ShowSeconds	bool	Gibt an, ob die Anzeige von Sekunden aktiviert ist	False
Opacity	float	Gibt die Opazität der Uhr an Der Wert der Opazität sollte zwischen 0 - 1 liegen	False
TimeSource	string	Gibt die Zeitquelle der Uhr an	L/S
Rotation	short	Gibt den Drehwinkel der Uhr an	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt der Uhr an	L/S
Visibility	bool	Gibt die Sichtbarkeit der Uhr an	L/S
Width	uint	Gibt die Breite der Uhr an	L/S
TabIndex	ushort	Gibt den Tab-Index der Uhr an	L/S
Title	HmiTextPart	Gibt den Titel der Uhr an	L
ToolTipText	MultiLingualText	Gibt den Tooltip-Text der Uhr an	L/S
EventHandlers	HmiClockEventHandlerComposition	Gibt den Ereignis-Handler der Uhr an	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
DialLabelFont	HmiFontPart	Gibt die Schriftart von Beschriftungen auf dem Zifferblatt der Uhr an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften einer Uhr zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ClockPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiClock clock = screenitems.Create<HmiClock>("Default Value334554");
    // To access basic properties of clock
    //Name
    var name = clock.Name;
    clock.Name = "Default Value";
    //AlternateBackColor
    var altbackcolor = clock.AlternateBackColor;
    clock.AlternateBackColor = Color.Beige;
    //Height
    var height = clock.Height;
    clock.Height = 100;
    // To access multilingual property of clock
    //ToolTipText
    var tooltip = clock.ToolTipText;
    var tooltiptext = clock.ToolTipText.Items[0].Text;
    clock.ToolTipText.Items[0].Text = <body><p>"TestforMultilingualProperty"</p></body>;
    // To access other typical properties
    //DialLabelFont
    var dialfont = clock.DialLabelFont;
    dialfont.Italic = false;
    var fontname = dialfont.FontName;
    dialfont.FontName = HmiFontName.SimSun;
    var size = dialfont.Size;
    dialfont.Size = 10.2f;
    var stike = dialfont.StrikeOut;
    dialfont.StrikeOut = true;
    var underline = dialfont.Underline;
    dialfont.Underline = false;
    dialfont.Bold = true;
    //ShowHours
    var showhour = clock.ShowHours;
    clock.ShowHours = false;
}
```

Auf Eigenschaften eines Zeigerinstruments zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Zeigerinstrument-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des Zeigerinstruments an	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Zeigerinstruments an	L/S
Authorization	string	Gibt die Autorisierung des Zeigerinstruments an	L
BackgroundColor	Color	Gibt die Hintergrundfarbe des Zeigerinstruments an	L/S
BorderColor	Color	Gibt die Rahmenfarbe des Zeigerinstruments an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Zeigerinstruments an	L/S
RelativeToOrigin	bool	Gibt an, ob der relative Ursprungswert des Zeigerinstruments angezeigt wird oder nicht	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Zeigerinstruments an	L
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
Title	HmiTextPart	Gibt den Titel des Zeigerinstruments an	L
Font	HmiFontPart	Gibt die Schriftart des Zeigerinstruments an	L/S
Label	HmiTextPart	Gibt die Beschriftung des Zeigerinstruments an	L
Height	uint	Gibt die Höhe des Zeigerinstruments an	L/S
Top	int	Gibt die X-Position des Zeigerinstruments an	L/S
Left	int	Gibt die Y-Position des Zeigerinstruments an	L/S
Name	string	Gibt den Namen des Zeigerinstruments an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
NormalRangeColor	Color	Gibt die Farbe des Normalbereichs des Zeigerinstruments an	L/S
OriginValue	double	Gibt den Ursprungswert des Zeigerinstruments an	L/S
OutputFormat	string	Gibt das Ausgabeformat des Zeigerinstruments an Der Wert des Ausgabeformats sollte zwischen 1 - 128 liegen	L/S
PeakIndicators	HmiPeakIndicator	Gibt die Spitzenwert-Anzeige des Zeigerinstruments an	L/S
ProcessValue	string	Gibt den Prozesswert des Zeigerinstruments an	L/S
ProcessValueIndicatorBackColor	Color	Gibt die Hintergrundfarbe der Prozesswertanzeige an	L/S
ProcessValueIndicatorForeColor	Color	Gibt die Vordergrundfarbe der Prozesswertanzeige an	L/S
ProcessValueIndicatorMode	HmiProcessIndicator-Mode	Gibt den Moduswert der Prozesswertanzeige an	L/S
Rotation	short	Gibt den Drehwinkel des Zeigerinstruments an	L/S
VisualizeQuality	bool	Gibt die Verbindungsqualität des Zeigerinstruments an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Zeigerinstruments an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Zeigerinstruments an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Opacity	float	Gibt die Opazität des Zeigerinstruments an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
RotationCenter-Placement	HmiRotationCenter-Placement	Gibt den Drehpunkt des Zeigerinstruments an	L/S
ScaleBackgroundColor	Color	Gibt die Hintergrundfarbe der Skala des Zeigerinstruments an	L/S
ScaleForegroundColor	Color	Gibt die Vordergrundfarbe der Skala des Zeigerinstruments an	L/S
TrendIndicatorColor	Color	Gibt an, ob die Kurvenanzeige des Zeigerinstruments dargestellt werden soll	L/S
Visible	bool	Gibt die Sichtbarkeit des Zeigerinstruments an	L/S
Width	uint	Gibt die Breite des Zeigerinstruments an	L/S
TabIndex	ushort	Gibt den Tab-Index des Zeigerinstruments an	L/S
ShowTrendIndicator	bool	Gibt an, ob die Kurvenanzeige eines Zeigerinstruments angezeigt wird	L/S
ToolTipText	string	Gibt den Tooltip-Text des Zeigerinstruments an	L/S
EventHandlers	HmiGaugeEventHandlerComposition	Gibt den Ereignis-Handler des Zeigerinstruments an	L
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
CurvedScale	HmiCurvedScalePart	Gibt die Skala des Zeigerinstruments an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften eines Zeigerinstruments zuzugreifen, ändern Sie folgenden Programmcode:

```
private void GaugeScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiGauge gauge = screenitems.Create<HmiGauge>("Default Value334554");
    //Name
    var name = gauge.Name;
    gauge.Name = "Default Value";
    //RotationalCenterPlacement
    var rotation = gauge.RotationCenterPlacement;
    gauge.RotationCenterPlacement = HmiRotationCenterPlacement.AbsoluteToContainer;
    //Width
    var width = gauge.Width;
    gauge.Width = 100;
    //ToolTipText
    var tooltip = gauge.ToolTipText;
    var tooltiptext = gauge.ToolTipText.Items[0].Text;
    gauge.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //Font
    var title = gauge.Title;
    title.Text.items[0].Text = "<body><p>teststing</p></body>";
    title.Visible = false;
    title.ForeColor = Color.Black;
    var font = title.Font;
    var italic = font.Italic;
    font.Italic = false;
    var fontname = font.Name;
    font.Name = HmiFontName.SimSun;
    var size = font.Size;
    font.Size = 10.2f;
    var stike = font.StrikeOut;
    font.StrikeOut = true;
    var underline = font.Underline;
    font.Underline = false;
    font.Weight = HmiFontWeight.Bold;
}
```

Auf Eigenschaften eines EA-Felds zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden bei Bildobjekten von EA-Feld-Eigenschaften unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
AlternateBackColor	Color	Gibt die alternative Hintergrundfarbe des EA-Felds an	L/S
Enabled	Bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des EA-Felds an	L/S
Authorization	String	Gibt die Autorisierung des EA-Felds an	L
BackColor	Color	Gibt die Hintergrundfarbe des EA-Felds an	L/S
BorderColor	Color	Gibt die Rahmenfarbe des EA-Felds an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des EA-Felds an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des EA-Felds an	L
Font	HmiFontPart	Gibt die Schrift des EA-Felds an	L/S
ForegroundColor	Color	Gibt die Vordergrundfarbe des EA-Felds an	L/S
Height	uint	Gibt die Höhe des EA-Felds an	L/S
HorizontalTextAlignment	HmiHorizontalAlignment	Gibt die horizontale Ausrichtung des EA-Felds an	L/S
IOFieldType	HmiIOFieldType	Gibt den Modus des EA-Felds an	L/S
Top	int	Gibt die Y-Position des EA-Felds an	L/S
Left	int	Gibt die X-Position des EA-Felds an	L/S
Rotation Angle	short	Gibt die Drehung des EA-Felds an	L/S
VisualizeQuality	bool	Gibt die Visualisierungsqualität des EA-Felds an	L/S
TextTrimming	HmiTextTrimming	Gibt den zu kürzenden Text des EA-Felds an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des EA-Felds an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des EA-Felds an	L/S
Opacity	float	Gibt die Opazität des EA-Felds an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen des EA-Felds an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
OutputFormat	string	Gibt das Ausgabeformat des EA-Felds an	L/S
Padding	HmiPaddingPart	Gibt den Innenabstand (Padding) des EA-Felds an	L
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des EA-Felds an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
ProcessValue	string	Gibt den Prozesswert des EA-Felds an	L/S
TabIndex	ushort	Gibt den Tab-Index des EA-Felds an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des EA-Felds an	L/S
VerticalTextAlign-ment	HmiVerticalAlign-ment	Gibt die vertikale Ausrichtung des EA-Felds an	L/S
Visible	bool	Gibt die Sichtbarkeit des EA-Felds an	L/S
Width	uint	Gibt die Breite des EA-Felds an	L/S
EventHandlers	HmiIOFieldEventHan- dle rComposition	Gibt den Ereignis-Handler des EA-Felds an	L
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
InputBehavior	HmiInputBehaviorPart	Gibt die Reaktion des EA-Felds auf Eingaben an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften von EA-Feld-Bildobjekten zuzugreifen, ändern Sie folgenden Programmcode:

```
private void IOFieldScreenItemsPropertiesAccess(Project project)
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiIOField iofield = screenitems.Create<HmiIOField>("Default Value354");
    //Name
    var name = iofield.Name;
    iofield.Name = "DefaultName";
    //AlternateBackColor
    var altbackcolor = iofield.AlternateBackColor;
    iofield.AlternateBackColor = Color.Beige;
    //Height
    var height = iofield.Height;
    iofield.Height = 100;
    //ToolTipText
    string culture = "en-US";
    Language lang = project.LanguageSettings.Languages.Find(new CultureInfo(culture));
    MultilingualText mltprop = iofield.ToolTipText;
    MultilingualTextItemComposition textItemComp = mltprop.Items;
    MultilingualTextItem mlttextitem = textItemComp.Find(lang);
    mlttextitem.Text = "CommentInEnglish";
    //Padding
    var padding = iofield.Padding;
    var bottom = padding.Bottom;
    padding.Bottom = 50;
    padding.Left = 60;
    padding.Right = 70;
    padding.Top = 50;
}
```

Auf Eigenschaften eines Listenfelds zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Listenfeld-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
AlternateBackgroundColor	Color	Gibt die alternative Hintergrundfarbe des Listenfelds an	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Listenfelds an	L/S
Authorization	string	Gibt die Autorisierung des Listenfelds an	L
BackgroundColor	Color	Gibt die Hintergrundfarbe des Listenfelds an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
BorderColor	Color	Gibt die Rahmenfarbe des Listenfelds an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Listenfelds an	L/S
Content	HmiContentPart	Gibt den Inhalt des Listenfelds an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Listenfelds an	L
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
Font	HmiFontPart	Gibt die Schriftart des Listenfelds an	L/S
ForegroundColor	Color	Gibt die Vordergrundfarbe des Listenfelds an	L/S
Top	int	Gibt die X-Position des Listenfelds an	L/S
Left	int	Gibt die Y-Position des Listenfelds an	L/S
Name	string	Gibt den Namen des Listenfelds an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Process Value	string	Gibt den Prozesswert des Listenfelds an	L/S
Rotation	short	Gibt den Drehwinkel des Listenfelds an	L/S
VisualizeQuality	bool	Gibt den Verbindungsstatus des Listenfelds an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Listenfelds an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Listenfelds an	L/S
SelectedItemHeight	uint16	Gibt die ausgewählte Objekthöhe des Listenfelds an	L/S
SelectionItems	IList<IHmiSelectionItemPart>	Gibt die ausgewählten Objekte des Listenfelds an	L
SelectionMode	HmiSelectionMode	Gibt den ausgewählten Modus des Listenfelds an	L/S
SelectorPosition	HmiHorizontalAlignment	Gibt die ausgewählte Position des Listenfelds an	L/S
Opacity	float	Gibt die Opazität des Listenfelds an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Listenfelds an	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Width	uint	Gibt die Breite des Listenfelds an	L/S
TabIndex	ushort	Gibt den Tab-Index des Listenfelds an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ToolTipText	string	Gibt den Tooltip-Text des Listenfelds an	L/S
EventHandlers	HmiListBoxEvent-Handler Composition	Gibt den Ereignis-Handler des Listenfelds an	L
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
Padding	HmiPaddingPart	Gibt den Innenabstand (Padding) des Listenfelds an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften eines Listenfelds zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ListBoxPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiListBox listBox = screenitems.Create<HmiListBox>("Default Value334554");
    //Name
    var name = listBox.Name;
    listBox.Name = "Default Value";
    //AlternateBackColor
    var altbackcolor = listBox.AlternateBackColor;
    listBox.AlternateBackColor = Color.Beige;
    //Height
    var height = listBox.Height;
    listBox.Height = 100;
    // To access multilingual property
    var tooltip = listBox.ToolTipText;
    var tooltiptext = listBox.ToolTipText.Items[0].Text;
    // To access other typical property of listBox
    //Font
    var font = listBox.Font;
    var italic = font.Italic;
    font.Italic = false;
    var fontname = font.Name;
    font.Name = HmiFontName.SimSun;
    var size = font.Size;
    font.Size = 10.2f;
    var stike = font.StrikeOut;
    font.StrikeOut = true;
    var underline = font.Underline;
    font.Underline = false;
    font.Weight = HmiFontWeight.Bold;
    //Selection Items
    var selectionItem = listBox.SelectionItems;
    var newselectionitem = selectionItem.Create();
    var graphic = newselectionitem.Graphic;
    newselectionitem.Graphic = "abcd";
    newselectionitem.IsSelected = false;
}
```

Auf Eigenschaften einer Optionsschaltfläche zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden bei Bildobjekten für Optionsschaltflächen unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
AlternateBackgroundColor	Color	Gibt die alternative Hintergrundfarbe der Optionsschaltfläche an	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe der Optionsschaltfläche an	L/S
Authorization	string	Gibt die Autorisierung der Optionsschaltfläche an	L
BackgroundColor	Color	Gibt die Hintergrundfarbe der Optionsschaltfläche an	L/S
BorderColor	Color	Gibt die Rahmenfarbe der Optionsschaltfläche an	L/S
BorderWidth	byte	Gibt die Rahmenbreite der Optionsschaltfläche an	L/S
Content	HmiContentPart	Gibt den Inhalt der Optionsschaltfläche an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Optionsschaltfläche an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
Font	HmiFontPart	Gibt die Schriftart der Optionsschaltfläche an	False
ForegroundColor	Color	Gibt die Vordergrundfarbe der Optionsschaltfläche an	L/S
Height	uint	Gibt die Höhe der Optionsschaltfläche an	L/S
Top	int	Gibt die X-Position der Optionsschaltfläche an	L/S
Left	int	Gibt die Y-Position der Optionsschaltfläche an	L/S
Name	string	Gibt den Namen der Optionsschaltfläche an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Process Value	string	Gibt den Prozesswert der Optionsschaltfläche an	L/S
Rotation	short	Gibt den Drehwinkel der Optionsschaltfläche an	L/S
VisualizeQuality	bool	Gibt den Verbindungsstatus der Optionsschaltfläche an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt der Optionsschaltfläche an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt der Optionsschaltfläche an	L/S
SelectionItemHeight	uint16	Gibt die ausgewählte Höhe der Optionsschaltfläche an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
SelectionItems	IList<IHmiSelectionItemPart>	Gibt die Auswahlobjekte der Optionsschaltfläche an	L
SelectorPosition	HmiHorizontalAlignment	Gibt die Auswahlposition der Optionsschaltfläche an	L
Opacity	float	Gibt die Opazität der Optionsschaltfläche an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt der Optionsschaltfläche an	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Width	uint	Gibt die Breite der Optionsschaltfläche an	L/S
TabIndex	ushort	Gibt den Tab-Index der Optionsschaltfläche an	L/S
ToolTipText	string	Gibt den Tooltip-Text der Optionsschaltfläche an	L/S
EventHandlers	HmiRadio buttonGroup EventHandlerComposition	Gibt den Ereignis-Handler der Optionsschaltfläche an	L
Padding	HmiPaddingPart	Gibt den Innenabstand (Padding) der Optionsschaltfläche an	L
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften einer Optionsschaltfläche zuzugreifen, ändern Sie folgenden Programmcode:

```
private void RadiobuttonPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiRadioButtonGroup radio = screenitems.Create<HmiRadioButtonGroup>("Default Value334554");
    // To access basic properties
    //Name
    var name = radio.Name;
    radio.Name = "Default Value";
    //AlternateBackColor
    var altbackcolor = radio.AlternateBackColor;
    radio.AlternateBackColor = Color.Beige;
    //Height
    var height = radio.Height;
    radio.Height = 100;
    // To access multilingual property
    //ToolTipText
    var tooltip = radio.ToolTipText;
    var tooltiptext = radio.ToolTipText.Items[0].Text;
    //radio.ToolTipText.Items[0].Text = "TestforMultilingualProperty";
    // To access other typical properties of radio button
    //Font
    var font = radio.Font;
    var italic = font.Italic;
    font.Italic = false;
    var fontname = font.Name;
    font.Name = HmiFontName.SimSun;
    var size = font.Size;
    font.Size = 10.2f;
    var stike = font.StrikeOut;
    font.StrikeOut = true;
    var underline = font.Underline;
    font.Underline = false;
    font.Weight = HmiFontWeight.Bold;
    //Selection Items
    var selectionItem = radio.SelectionItems;
    var newselectionitem = selectionItem.Create();
    var graphic = newselectionitem.Graphic;
    newselectionitem.Graphic = "abcd";
    newselectionitem.IsSelected = false;
}
```

Auf Eigenschaften eines Schiebereglers zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Schieberegler-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
AlternativeBackColor	Color	Gibt die alternative Hintergrundfarbe des Schiebereglers an	L/S
AlternativeBorderColor	Color	Gibt die alternative Rahmenfarbe des Schiebereglers an	L/S
Authorization	string	Gibt die Autorisierung des Schiebereglers an	L
BackgroundColor	Color	Gibt die Hintergrundfarbe des Schiebereglers an	L/S
BorderColor	Color	Gibt die Rahmenfarbe des Schiebereglers an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Schiebereglers an	L/S
BarMode	HmiBarMode	Gibt den Balkenmodus des Schiebereglers an	L/S
RelativeToOrigin	bool	Gibt den Relativwert zum Ursprungswert des Schiebereglers an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Schiebereglers an	L
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	False
Title	HmiTextPart	Gibt den Titel des Schiebereglers an	L
Font	HmiFontPart	Gibt die Schriftart des Schiebereglers an	L/S
Label	HmiTextPart	Gibt die Beschriftung des Schiebereglers an	L
Height	uint	Gibt die Höhe des Schiebereglers an	L/S
Top	int	Gibt die X-Position des Schiebereglers an	L/S
Left	int	Gibt die Y-Position des Schiebereglers an	L/S
Name	string	Gibt den Namen des Schiebereglers an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
NormalRangeColor	Color	Gibt die Farbe des Normalbereichs des Schiebereglers an	L/S
OriginValue	double	Gibt den Ursprungswert des Schiebereglers an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
OutputFormat	string	Gibt das Ausgabeformat des Schiebereglers an Der Wert des Ausgabeformats sollte zwischen 1 - 128 liegen	L/S
PeakIndicators	HmiPeakIndicator	Gibt die Spitzenwert-Anzeige des Schiebereglers an	L/S
Process Value	string	Gibt den Prozesswert des Schiebereglers an	L/S
ProcessValueIndicator-BackColor	Color	Gibt die Hintergrundfarbe der Prozesswertanzeige des Schiebereglers an	L/S
ProcessValueIndicator-ForeColor	Color	Gibt die Vordergrundfarbe der Prozesswertanzeige des Schiebereglers an	L/S
ProcessValueIndicator-Mode	HmiProcessIndicatorMode	Gibt den Modus der Prozesswertanzeige des Schiebereglers an	L/S
Rotation	short	Gibt den Drehwinkel des Schiebereglers an	L/S
VisualizeQuality	bool	Gibt die Verbindungsqualität des Schiebereglers an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Schiebereglers an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Schiebereglers an	L/S
Opacity	float	Gibt die Opazität des Schiebereglers an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Schiebereglers an	L/S
ScaleBackgroundColor	Color	Gibt die Hintergrundfarbe der Skala des Schiebereglers an	L/S
ScaleForegroundColor	Color	Gibt die Vordergrundfarbe der Skala des Schiebereglers an	L/S
TrendIndicatorColor	bool	Gibt an, ob die Kurvenanzeige des Schiebereglers eingeblendet werden soll	L/S
Visible	bool	Gibt die Sichtbarkeit des Schiebereglers an	L/S
Width	uint	Gibt die Breite des Schiebereglers an	L/S
ValuePosition	HmiSimplePosition	Gibt die Wertposition des Schiebereglers an	L/S
WriteDuring-Change(Write process value Immediatly)	bool	Gibt an, ob „Prozesswert schreiben“ aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index des Schiebereglers an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ShowTrendIndicator	bool	Gibt die Farbe der Kurvenanzeige des Schiebereglers an	L/S
ToolTipText	string	Gibt den Tooltip-Text des Schiebereglers an	L/S
ShowValue	bool	Gibt an, ob der Wert eines Schiebereglers angezeigt wird	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
EventHandlers	HmiSliderEventHandlerComposition	Gibt den Ereignis-Handler des Zeigerinstruments an	L
ThumbBackColor	color	Gibt die Hintergrundfarbe der Miniaturansicht des Schiebereglers an	L/S
ThumbForeColor	color	Gibt die Vordergrundfarbe der Miniaturansicht des Schiebereglers an	L/S
StraightScale	HmiStraightScalePart	Gibt die geradlinige Skala des Schiebereglers an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf Eigenschaften eines Schiebereglers zuzugreifen, ändern Sie folgenden Programmcode:

```
private void SliderPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiSlider slider = screenitems.Create<HmiSlider>("Default Value334554");
    // To access basic properties of slider
    //Name
    var name = slider.Name;
    slider.Name = "Default Value";
    //RotationalCenterPlacement
    var rotation = slider.RotationCenterPlacement;
    slider.RotationCenterPlacement = HmiRotationCenterPlacement.AbsoluteToContainer;
    //Width
    var width = slider.Width;
    slider.Width = 100;
    // To access multilingual property of slider
    //ToolTipText
    var tooltip = slider.ToolTipText;
    var tooltiptext = slider.ToolTipText.Items[0].Text;
    //slider.ToolTipText.Items[0].Text = "TestforMultilingualProperty";
    // To access other typical properties of slider
    //Font
    var title = slider.Title;
    //title.Text.Items[0].Text = "teststing";
    title.Visible = false;
    title.ForeColor = Color.Black;
    var font = title.Font;
    var italic = font.Italic;
    font.Italic = false;
    var fontname = font.Name;
    font.Name = HmiFontName.SimSun;
    var size = font.Size;
    font.Size = 10.2f;
    var stike = font.StrikeOut;
    font.StrikeOut = true;
    var underline = font.Underline;
    font.Underline = false;
    font.Weight = HmiFontWeight.Bold;
}
```

Auf Eigenschaften eines Schalters zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Schalter-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
AlternateBackgroundColor	Color	Gibt die alternative Hintergrundfarbe des Schalters an	L/S
AlternateBorderColor	Color	Gibt die alternative Rahmenfarbe des Schalters an	L/S
IsAlternateState	bool	Gibt den alternativen Zustand des Schalters an	L/S
Authorization	string	Gibt die Autorisierung des Schalters an	L
BackgroundColor	Color	Gibt die Hintergrundfarbe des Schalters an	L/S
BorderColor	Color	Gibt die Rahmenfarbe des Schalters an	L/S
BorderWidth	byte	Gibt die Rahmenbreite des Schalters an	L/S
Contents	HmiContentPart	Gibt den Inhalt des Schalters an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Schalters an	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
Font	HmiFontPart	Gibt die Schriftart des Schalters an	L/S
ForegroundColor	Color	Gibt die Vordergrundfarbe des Schalters an	L/S
Height	uint	Gibt die Höhe des Schalters an	L/S
Top	int	Gibt die Y-Position des Schalters an	L/S
Left	int	Gibt die X-Position des Schalters an	L/S
Rotation	short	Gibt die Drehung des Schalters an	L/S
Text	MultilingualText	Gibt den Text des Schalters an	L
VisualizeQuality	bool	Gibt die Visualisierungsqualität des Schalters an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des Schalters an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des Schalters an	L/S
Opacity	float	Gibt die Opazität des Schalters an Der Wert der Opazität sollte zwischen 0 - 1 liegen	L/S
Name	string	Gibt den Namen des Schalters an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Padding	HmiPaddingPart	Gibt den Innenabstand (Padding) des Schalters an	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Schalters an	L/S
TabIndex	ushort	Gibt den Tab-Index des Schalters an	L/S
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Schalters an	L/S
Visiblity	bool	Gibt die Sichtbarkeit des Schalters an	L/S
Width	uint	Gibt die Breite des Schalters an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
EventHandlers	HmiToggleswitchEventHandlerComposition	Gibt den Ereignis-Handler des Schalters an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S
Graphic	string	Gibt die Grafik des Schalters an Der Wert der Grafik sollte zwischen 1 - 128 liegen	L/S
AlternateGraphic	string	Gibt die Grafik des gedrückten Schalters an Der Wert der alternativen Grafik sollte zwischen 1 - 128 liegen	L/S
AlternateText	MultiLingualText	Gibt den Text eines gedrückten Schalters an	L
Padding	HmiPaddingPart	Gibt den Innenabstand (Padding) des Schalters an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften von Schalter-Bildobjekten zuzugreifen, ändern Sie folgenden Programmcode:

```
private void switchScreenItemsPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.ScreenItems;
    HmiToggleSwitch toggleswitch = screenitems.Create<HmiToggleSwitch>("Default Value354");
    //Name
    var name = toggleswitch.Name;
    toggleswitch.Name = "Default Value";
    //AlternateBackColor
    var altbackcolor = toggleswitch.AlternateBackColor;
    toggleswitch.AlternateBackColor = Color.Beige;
    //Height
    var height = toggleswitch.Height;
    toggleswitch.Height = 100;
    //ToolTipText
    var tooltip = toggleswitch.ToolTipText;
    var tooltiptext = toggleswitch.ToolTipText.Items[0].Text;
    toggleswitch.ToolTipText.Items[0].Text = "TestforMultilingualProperty";
    //Font
    var font = toggleswitch.Font;
    var italic = font.Italic;
    font.Italic = false;
    var fontname = font.Name;
    font.Name = HmiFontName.SimSun;
    var size = font.Size;
    font.Size = 10.2f;
    var stike = font.StrikeOut;
    font.StrikeOut = true;
    var underline = font.Underline;
    font.Underline = false;
    font.Weight = HmiFontWeight.Bold;
}
```

Auf Eigenschaften eines symbolischen EA-Felds zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei einem symbolischen HMI-EA-Feld unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
Authorization	string	Gibt die Autorisierung des symbolischen EA-Felds an	L/S
CurrentQuality	HmiQuality	Gibt die Verbindungsqualität des symbolischen EA-Felds an	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
Dynamizations	DynamizationBaseComposition	Gibt die Dynamisierungen des symbolischen EA-Felds an	L
Enabled	bool	Gibt an, ob "Bedienelement erlauben" aktiviert ist	L/S
EventHandlers	HmiSymbolicIOFieldEventHandlerComposition	Gibt den Ereignis-Handler des symbolischen EA-Felds an	L
Height	uint	Gibt die Höhe des symbolischen EA-Felds an	L
IOFieldType	HmiIOFieldType	Gibt den Feldtyp des symbolischen EA-Felds an	L/S
ProcessValue	String	Gibt den Prozesswert des symbolischen EA-Felds an	L/S
ResourceList	String	Gibt die Textliste und Grafikliste für das symbolische EA-Feld an	L/S
Selected Index	Int32	Gibt den ausgewählten Index des symbolischen EA-Felds an	L
SelectionBackColor	Color	Gibt die ausgewählte Hintergrundfarbe des symbolischen EA-Felds an	L/S
SelectionForeColor	Color	Gibt die ausgewählte Vordergrundfarbe des symbolischen EA-Felds an	L/S
Text	MultilingualText	Gibt den Text des symbolischen EA-Felds an	L/S
Left	int	Gibt die X-Achsenposition des symbolischen EA-Felds an	L/S
Name	String	Gibt den Namen des symbolischen EA-Felds an	L/S
Opacity	float	Gibt den Opazitätswert des symbolischen EA-Felds an	L/S
PropertyEventHandlers	PropertyEventHandlerComposition	Gibt die Eigenschafts-Ereignis-Handler des symbolischen EA-Felds an	L
RequireExplicitUnlock	bool	Gibt an, ob der Wert der expliziten Entriegelung angezeigt wird oder nicht	L/S
RotationAngle	short	Gibt den Drehwinkel des symbolischen EA-Feld an	L/S
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des symbolischen EA-Felds an	L/S
RotationCenterX	float	Gibt den X-Drehpunkt des symbolischen EA-Felds an	L/S
RotationCenterY	float	Gibt den Y-Drehpunkt des symbolischen EA-Felds an	L/S
ShowFocusVisual	bool	Gibt an, ob "Blickfeld anzeigen" aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index des Kontrollkästchens an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugriff
ToolTipText	MultilingualText	Gibt den im Tooltip zu erwähnenden Text an	L
Top	int	Gibt die Y-Position des symbolischen EA-Felds an	L/S
Visible	bool	Gibt die Sichtbarkeit des symbolischen EA-Felds an	L/S
VisualizeQuality	bool	Gibt die visuelle Qualität des symbolischen EA-Felds an	L/S
Width	uint	Gibt die Breite des symbolischen EA-Felds an	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die grundlegenden Eigenschaften eines symbolischen EA-Felds zuzugreifen, ändern Sie folgenden Programmcode:

```
private void SymbolicIOFieldPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreenComposition screens = hmiSoftware.Screens;
    HmiScreen hmiScreen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenItems = hmiScreen.ScreenItems
    HmiSymbolicIOField objSymbField =
    screenItems.Create<HmiSymbolicIOField>("Screen_object_1");
    //Left
    objSymbField.Left = 1;
    //ToolTipText
    objSymbField.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
}
```

Auf Eigenschaften eines Touchbereichs zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei einem HMI-Touchbereich unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Authorization	string	Gibt die Autorisierung des HMI-Touchbereichs an	R/W
Dynamizations	DynamizationBaseComposition	Gibt die Dynamisierung des HMI-Touchbereichs an	R
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	R/W
EventHandlers	HmiTouchAreaEventHandlerComposition	Gibt den Ereignis-Handler des HMI-Touchbereichs an	R
Height	uint	Gibt die Höhe des HMI-Touchbereichs an	R/W
BackColor	Color	Gibt die Hintergrundfarbe des HMI-Touchbereichs an	R/W
Left	int	Gibt die X-Position des HMI-Touchbereichs an	R/W
Name	string	Gibt den Namen des HMI-Touchbereichs an	R/W
PropertyEventHandlers	PropertyEventHandlerComposition	Gibt den Eigenschafts-Ereignis-Handler des HMI-Touchbereichs an	R
RequireExplicitUnlock	bool	Gibt an, ob eine explizite Entriegelung erforderlich ist	R/W
TabIndex	ushort	Gibt den Tab-Index des HMI-Touchbereichs an	R/W
Top	int	Gibt die Y-Position des HMI-Touchbereichs an	R/W
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	R/W
Width	uint	Gibt die Breite des HMI-Touchbereichs an	R/W

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die grundlegenden Eigenschaften eines HMI-Touchbereichs zuzugreifen, ändern Sie folgenden Programmcode:

```
private void HMITouchAreaPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreenComposition screens = hmiSoftware.Screens;
    HmiScreen screen = hmiSoftware.Screens.Find("Screen_1");
    HmiScreenItemBaseComposition screenItems = hmiScreen.ScreenItems;
    HmiTouchArea objTouchArea = screenItems.Create<HmiTouchArea>("Screen_object_1");
    //Left
    objTouchArea.Left = 1;
    //Backcolor
    objTouchArea.Backcolor = Color.Red;
}
```

Bedienelemente (RT Unified)

Auf Eigenschaften einer Meldeanzeige zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Meldeanzeigen-Bildobjekten unterstützt.

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
UseAlarmColors	bool	Gibt die Farbe von Anwenderalarmen in der Meldeanzeige an	L/S
Filter	string	Gibt den Filter der Meldeanzeige an	L/S
AlwaysShowRecent	bool	Gibt an, dass das Bedienelement für die Anzeige der letzten Meldungen eingeblendet werden soll	L/S
AlarmSourceType	HmiAlarmSourceType	Gibt die Meldequelle der Meldeanzeige an	L/S
Caption	HmiTextPart	Gibt die Beschriftung an, die in der Meldeanzeige eingeblendet werden soll	L
WindowFlags	HmiWindowFlag	Gibt die Fenstereinstellung der Meldeanzeige an	L/S
Icon	string	Gibt das Symbol der Meldeanzeige an	L/S
Top	int	Gibt die Y-Position der Meldeanzeige an	L/S
Left	int	Gibt die X-Position der Meldeanzeige an	L/S
Width	uint	Gibt die Breite der Meldeanzeige an	L/S
Height	uint	Gibt die Höhe der Meldeanzeige an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Meldeanzeige an	L
Name	string	Gibt den Namen der Meldeanzeige an	L/S
Visible	bool	Gibt die Sichtbarkeit der Meldeanzeige an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index der Meldeanzeige an	L/S
ToolBar	HmiToolBarPart	Gibt die Symbolleiste der Meldeanzeige an	L
StatusBar	HmiStatusBarPart	Gibt die Statusleiste der Meldeanzeige an	L
EditMode	HmiEditMode	Gibt den Bearbeitungsmodus der Meldeanzeige an	L/S
TimeZone	int	Gibt die Zeitzone der Meldeanzeige an	L/S
AlarmDefinitionView-Setup	HmiVisibleAlarms	Gibt die angezeigten Alarme der Meldeanzeige an	L/S
ActiveAlarmsView-Setup	HmiVisibleAlarms	Gibt die aktuellen Meldungen der Meldeanzeige an	L/S
SuppressFlashing	bool	Gibt an, dass das Blinken der Meldeanzeige unterdrückt werden soll	L/S
AcknowledgmentFlashingRate	HmiFlashingRate	Gibt die Meldungen der Meldeanzeige an, über die die Blinkfrequenz quittiert wird	L/S
ResetFlashingRate	HmiFlashingRate	Gibt das Rücksetzen der Blinkfrequenz der Meldeanzeige an	L/S
DefaultSortDirection	HmiSortDirection	Gibt die standardmäßige Sortierrichtung an	L/S
BackColor	Color	Gibt die Hintergrundfarbe der Meldeanzeige an	L/S
AlarmView	HmiDataGridView-Part	Gibt die Meldeansicht der Meldeanzeige an	TRUE
EventHandlers	HmiAlarmControllerEventHandler-Composition	Gibt die Ereignis-Handler der Meldeanzeige an	L
CaptionColor	Color	Gibt die Farbe des Titels der Meldeanzeige an	L/S
AlarmStatisticsView	HmiDataGridView-Part	Gibt die Anzeige der Meldestatistik an	L
ShowFocusVisual	Boolean	Gibt an, ob das Blickfeld für eine Meldeanzeige eingeblendet werden soll	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften einer Meldeanzeige zuzugreifen, ändern Sie folgenden Programmcode:

```
private void AccessAlarmControlProperties()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    var screen = hmiSoftware.Screens;
    var createdscreen = screen.Create("TestScreen_1");
    var hmialarmcontrol=
Createdscreen.ScreenItems.Create<HmiAlarmControl>("HmiAlarmControl_1");
    // Basic properties Access
    // Width
    var autoplay = hmialarmcontrol.Width;
    hmialarmcontrol.Width = 100;
    // BackColor
    var backcolor = hmialarmcontrol.BackColor;
    hmialarmcontrol.BackColor = Color.Beige;
    // Name
    var name = hmialarmcontrol.Name;
    hmialarmcontrol.Name = "DefaultName";
    // MultiLingual Property Access
    // Access caption
    var caption = hmialarmcontrol.Caption;
    IHmiWindowFeature.Caption.Text.Items[0].Text = "<body><p>TestforMultilingualProperty</
body></p>";
    // Access other typical properties
    var statusBar = hmialarmcontrol.StatusBar;
    var backColor = statusBar.BackColor;
    statusBar.BackColor = Color.Aqua;
    var enabled = statusBar.Enabled;
    statusBar.Enabled = true;

    var visible = statusBar.Visible;
    statusBar.Visible = true;
}
```

Auf Eigenschaften einer f(x)-Kurvenanzeige zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei einer f(x)-Kurvenanzeige unterstützt:

Name der Eigen-schaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ShowStatisticRulers	bool	Gibt an, dass die Statistik-Lineale der f(x)-Kurvenanzeige eingeblendet werden sollen	L/S
AreaSpacing	ushort	Gibt die Bereichsfläche in der f(x)-Kurvenan-zeige an	L/S
Online	bool	Gibt die Onlineverbindung an	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ExtendRulerToAxis	bool	Gibt an, dass das Lineal zusätzlich zur Achse in der f(x)-Kurvenanzeige eingeblendet werden soll	L/S
Caption	MultilingualProperty	Gibt den Text an, der im Titel des Bildfensters oder Fensterbedienelements (Label) angezeigt werden soll	L/S
WindowFlags	HmiWindowFlag	Gibt die Fensterkonfiguration an, z. B. ShowCaption, ShowBorder, AlwaysOnTop	L/S
Icon	string	Gibt das Symbol im Steuerfenster an	L/S
Top	Int	Gibt den Wert der Y-Koordinaten der f(x)-Kurvenanzeige an	L/S
Left	int	Gibt den Wert der X-Koordinaten der f(x)-Kurvenanzeige an	L/S
Width	uint	Gibt den Breitenwert der f(x)-Kurvenanzeige an	L/S
Height	uint	Gibt den Höhenwert der f(x)-Kurvenanzeige an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der f(x)-Kurvenanzeige an	L
Name	string	Gibt den Namen der f(x)-Kurvenanzeige an	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
TabIndex	ushort	Bildobjekte, die einen Tab-Index 0 vorgeben, sind nicht Teil der Tab-Reihenfolge	L/S
ToolBar	HmiToolBarPart	Gibt die Symbolleiste der f(x)-Kurvenanzeige an	L
StatusBar	HmiStatusBarPart	Gibt die Statusleiste in der f(x)-Kurvenanzeige an	L
	HmiFontPart	Gibt die Schriftart der f(x)-Kurvenanzeige an	L
TimeZone	int	Gibt die Zeitzone an	L/S
ShowRuler	bool	Gibt das Lineal an	False
ShiftAxes	bool	Gibt die Verschiebung von Achsen der f(x)-Kurvenanzeige an	L/S
BackColor	Color	Gibt die Hintergrundfarbe der f(x)-Kurvenanzeige an	L/S
Legend	HmiLegendPart	Gibt die Legende an	L
EventHandlers	HmiFunctionTrendControlEventHandlerComposition	Gibt die Ereignis-Handler der f(x)-Kurvenanzeige an	L
FunctionTrendAreas	HmiFunctionTrendAreaPartComposition	Gibt die Bereiche einer f(x)-Kurvenanzeige an	L
CaptionColor	Color	Gibt die Farbe des Titels der f(x)-Kurvenanzeige an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf Eigenschaften einer f(x)-Kurvenanzeige zuzugreifen, ändern Sie folgenden Programmcode:

```
private void FunctionTrendControlPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen screen = hmiSoftware.Screens.Create("Screen_164");
    HmiFunctionTrendControl hmifunctiontrendcontrol =
    screen.ScreenItems.Create<HmiFunctionTrendControl>("CTrendControl");
    // To access basic properties of function trend control
    //Width
    var autoplay = hmifunctiontrendcontrol.Width;
    hmifunctiontrendcontrol.Width = 100;
    //BackColor
    var backcolor = hmifunctiontrendcontrol.BackColor;
    hmifunctiontrendcontrol.BackColor = Color.Beige;
    //Name
    var name = hmifunctiontrendcontrol.Name;
    hmifunctiontrendcontrol.Name = "DefaultName";
    // To access multilingual property
    //Caption
    var caption = hmifunctiontrendcontrol.Caption.Items[0].Text;
    hmifunctiontrendcontrol.Caption.Items[0].Text =
    "<body><p>TestforMultilingualProperty</p></body>";
    // To access other typical properties
    //Status bar
    var statusBar = hmifunctiontrendcontrol.StatusBar;
    var backColor = statusBar.BackColor;
    statusBar.BackColor = Color.Aqua;
    var enabled = statusBar.Enabled;
    statusBar.Enabled = true;
    var visible = statusBar.Visible;
    statusBar.Visible = true;
}
```

Programmcode: TrendArea in FunctionTrendControl using part

Um in einer f(x)-Kurvenanzeige einer Hmi-Funktion, die ein Part-Objekt verwendet, einen Kurvenbereich zu erstellen, ändern Sie folgenden Programmcode:

```
private void TrendAreaCreateUsingPart()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    var screen = hmiSoftware.Screens;
    var createdscreen = screen.Create("TestScreen_1");
    HmiFunctionTrendControl hmifunctiontrendcontrol =
    hmiSoftware.Screens[0].ScreenItems.Create<HmiFunctionTrendControl>("CTrendControl_1");
    HmiFunctionTrendAreaPart part = hmifunctiontrendcontrol.FunctionTrendAreas.Create("part1");
}
```

Um die f(x)-Kurvenanzeige einer Hmi-Funktion, die ein Part-Objekt verwendet, zu durchsuchen, ändern Sie folgenden Programmcode:

```
private void TrendAreaPartSearch()
{
    HmiFunctionTrendAreaPart part = hmifunctiontrendcontrol.FunctionTrendAreas.Find("part1");
}
```

Um in einer f(x)-Kurvenanzeige einer Hmi-Funktion, die ein Part-Objekt verwendet, einen Kurvenbereich zu prüfen, ändern Sie folgenden Programmcode:

```
private void IsTrendAreaExist()
{
    bool bPresent = hmifunctiontrendcontrol.FunctionTrendAreas.Contains(part);
}
```

Auf Eigenschaften eines Media Players zugreifen (RT Unified)**Einleitung**

Die folgenden Eigenschaften werden bei Media-Player-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Url	string	Gibt die URL des Media Players an	L/S
AutoPlay	bool	Gibt die Funktion Autoplay des Media Players an	L/S
VideoOutput	HmiVideoOutput	Gibt die Videoausgabe des Media Players an	L/S
BackColor	Color	Gibt die Hintergrundfarbe der Tabellenanzeige an	L/S
Caption	MultilingualProperty	Gibt den Titel der Tabellenanzeige an	L/S
WindowFlags	HmiWindowFlag	Gibt die Fenster-Eigenschaften der Tabellenanzeige an	L/S
Icon	string	Gibt das Symbol der Tabellenanzeige an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Top	int	Gibt die X-Position der Tabellenanzeige an	L/S
Left	int	Gibt die Y-Position der Tabellenanzeige an	L/S
Width	uint	Gibt die Breite der Tabellenanzeige an	L/S
Height	uint	Gibt die Höhe der Tabellenanzeige an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Media Players an	True
Name	string	Gibt den Namen der Tabellenanzeige an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index der Tabellenanzeige an	L/S
ToolBar	HmiToolBarPart	Gibt die Symbolleiste der Tabellenanzeige an	L
StatusBar	HmiStatusBarPart	Gibt die Statusleiste der Tabellenanzeige an	L
EventHandlers	HmiMediaControlEventHandlerComposition	Gibt den Ereignis-Handler der Tabellenanzeige an	L
CaptionColor	Color	Gibt die Farbe des Titels der Tabellenanzeige an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die grundlegenden Eigenschaften eines Media Players zuzugreifen, ändern Sie folgenden Programmcode:

```
private void MediaPlayerPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    var screen = hmiSoftware.Screens;
    var createdscreen = screen.Create("TestScreen_156");
    var hmimediacontrol =
    createdscreen.ScreenItems.Create<HmiMediaControl>("HmiMediaControl_1");
    // To access basic properties
    //Autoplay
    var autoplay = hmimediacontrol.AutoPlay;
    hmimediacontrol.AutoPlay = true;
    //BackColor
    var backcolor = hmimediacontrol.BackColor;
    hmimediacontrol.BackColor = Color.Beige;
    //Name
    var name = hmimediacontrol.Name;
    hmimediacontrol.Name = "DefaultName";
    // To access multilingual properties
    //Caption
    var caption = hmimediacontrol.Caption.Items[0].Text;
    hmimediacontrol.Caption.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    // To access other typical propeties
    var statusBar = hmimediacontrol.StatusBar;
    var backColor = statusBar.BackColor;
    statusBar.BackColor = Color.Aqua;
    var enabled = statusBar.Enabled;
    statusBar.Enabled = true;
    var visible = statusBar.Visible;
    statusBar.Visible = true;
}
```

Auf Eigenschaften einer Tabellenanzeige einstellen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Tabellenanzeigen-Bildobjekten unterstützt.

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Online	bool	Gibt die Online-Verbindung der Prozesssteuerung an	L/S
TimeStepSmoothingBase	HmiTimeRangeBase	Gibt das Zeitintervall für den Glättungs-Basiswert an	L/S
TimeStepSmoothingFactor	int	Gibt das Zeitintervall für den Glättungsfaktor an	L/S
BackColor	Color	Gibt die Hintergrundfarbe der Tabellenanzeige an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Caption	MultilingualProperty	Gibt den in der Prozesssteuerung zu erwähnenden Text an	L/S
WindowFlags	HmiWindowFlag	Gibt die Fenster-Eigenschaften der Tabellenanzeige an	L/S
Icon	string	Gibt das Symbol der Tabellenanzeige an	L/S
Top	int	Gibt die X-Position der Tabellenanzeige an	L/S
Left	int	Gibt die Y-Position der Tabellenanzeige an	L/S
Width	uint	Gibt die Breite der Tabellenanzeige an	L/S
Height	uint	Gibt die Höhe der Tabellenanzeige an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Tabellenanzeige an	L
Name	string	Gibt den Namen der Tabellenanzeige an Die Länge des Namens sollte zwischen 1 und 128 Zeichen betragen	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
TabIndex	ushort	Gibt den Tab-Index der Tabellenanzeige an	L/S
ToolBar	HmiToolBarPart	Gibt die Symbolleiste der Tabellenanzeige an	L
StatusBar	HmiStatusBarPart	Gibt die Statusleiste der Tabellenanzeige an	L
EditMode	HmiEditMode	Gibt den Bearbeitungsmodus der Tabellenanzeige an	L/S
TimeZone	int	Gibt die Zeitzone der Tabellenanzeige an	L/S
ProcessView	HmiDataGridViewPart	Gibt die Prozessansicht der Tabellenanzeige an	L
EventHandlers	HmiProcessControlEventHandlerComposition	Gibt den Ereignis-Handler der Tabellenanzeige an	L
CaptionColor	Color	Gibt die Farbe des Titels der Tabellenanzeige an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften einer Tabellenanzeige zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ProcessControlPropertiesAccess()
{
    // To access basic properties
    //Width
    var autoplay = hmiprocesscontrol.Width;
    hmiprocesscontrol.Width = 1;
    //BackColor
    var backcolor = hmiprocesscontrol.BackColor;
    hmiprocesscontrol.BackColor = Color.Beige;
    //Name
    var name = hmiprocesscontrol.Name;
    hmiprocesscontrol.Name = "DefaultName";
    // To access multilingual property
    //Caption
    var caption = hmiprocesscontrol.Caption.Items[0].Text;
    hmiprocesscontrol.Caption.Items[0].Text= "<body><p>TestforMultilingualProperty</p></body>";
    // To access other typical properties
    //status bar
    var statusBar = hmiprocesscontrol.StatusBar;
    var backColor = statusBar.BackColor;
    statusBar.BackColor = Color.Aqua;
    var enabled = statusBar.Enabled;
    statusBar.Enabled = true;
    var visible = statusBar.Visible;
    statusBar.Visible = true;
}
```

Auf Eigenschaften einer Parametersatzanzeige zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei einer Parametersatzanzeige unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ParameterSetType-Fixed	string		False
BackColor	Color	Gibt die Hintergrundfarbe der Parametersatzanzeige an	False
Caption	MultilingualProperty	Gibt den Text an, der im Titel des Bildfensters oder Fensterbedienelements (Label) angezeigt werden soll	False
WindowFlags	HmiWindowFlag	Gibt die Fensterkonfiguration an, z. B. ShowCaption, ShowBorder, AlwaysOn-Top	False
Icon	string	Gibt das Symbol im Steuerfenster an	False

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Top	int	Gibt den Wert der Y-Koordinaten der Parametersatzanzeige an	False
Left	int	Gibt den Wert der X-Koordinaten der Parametersatzanzeige an	False
Width	uint	Gibt den Breitenwert der Parametersatzanzeige an	False
Height	uint	Gibt den Höhenwert der Parametersatzanzeige an	False
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Parametersatzanzeige an	True
Name	string	Gibt den Namen der Parametersatzanzeige an	False
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	False
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	False
TabIndex	ushort	Bildobjekte, die einen Tab-Index 0 vorgeben, sind nicht Teil der Tab-Reihenfolge	False
ToolBar	HmiToolBarPart	Gibt die Symbolleiste der Parametersatzanzeige an	True
StatusBar	HmiStatusBarPart	Gibt die Statusleiste der Parametersatzanzeige an	True
EditMode	HmiEditMode	Gibt den Bearbeitungsmodus der Parametersatzanzeige an	False
TimeZone	int	Gibt die Zeitzone der Parametersatzanzeige an	False
ParameterView	HmiDataGridViewPart	Gibt die Parameteransicht der Parametersatzanzeige an	True
ForeColor	Color	Gibt die Vordergrundfarbe der Parametersatzanzeige an	False
SelectionBackColor	Color	Gibt die ausgewählte Hintergrundfarbe der Parametersatzanzeige an	False
SelectionForeColor	Color	Gibt die ausgewählte Vordergrundfarbe der Parametersatzanzeige an	False
Font	HmiFontPart	Gibt die Schriftart der Parametersatzanzeige an	True
EventHandlers	HmiDetailedParameterControlEventHandlerComposition	Gibt den Ereignis-Handler der Parametersatzanzeige an	True
CaptionColor	Color	Gibt die Farbe des Titels der Parametersatzanzeige an	False
HideDetails	bool	Gibt an, ob das Ausblenden von Details aktiviert ist	False
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	False

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf Eigenschaften einer Parametersatzanzeige zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ParameterSetControlPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen screen = hmiSoftware.Screens.Create("TestScreen");
    HmiDetailedParameterControl hmidetailedparametercontrol =
    screen.ScreenItems.Create<HmiDetailedParameterControl>("ParameterControl_1");
    // To access basic properties
    //Width
    var autoplay = hmidetailedparametercontrol.Width;
    hmidetailedparametercontrol.Width = 100;
    //BackColor
    var backcolor = hmidetailedparametercontrol.BackColor;
    hmidetailedparametercontrol.BackColor = Color.Beige;
    //Name
    var name = hmidetailedparametercontrol.Name;
    hmidetailedparametercontrol.Name = "Default";
    // To access multilingual property
    //Caption
    var caption = hmidetailedparametercontrol.Caption.Items[0].Text;
    hmidetailedparametercontrol.Caption.Items[0].Text=
    "<body><p>TestforMultilingualProperty</p></body>";
    // To access other typical property
    //Status bar
    var statusBar = hmidetailedparametercontrol.StatusBar;
    var backColor = statusBar.BackColor;
    statusBar.BackColor = Color.Aqua;
    var enabled = statusBar.Enabled;
    statusBar.Enabled = true;
    var visible = statusBar.Visible;
    statusBar.Visible = true;
}
```


Auf Eigenschaften eines Bildfensters zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Bildfenster-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
TabIntoWindow	bool	Gibt an, dass mit der Tab-Reihenfolge des dargestellten Bildfensters fortgefahren wird	L/S
Screen	string	Gibt das Bild des Bildfensters an	L/S
ScreenName	string	Gibt den Bildnamen des Bildfensters an	L
ScreenNumber	byte	Gibt die Bildnummer des Bildfensters an	L
System	byte	Gibt das System des Bildfensters an	L/S
CurrentZoomFactor	double	Gibt den aktuellen Zoomfaktor des Bildfensters an	L/S
VerticalScrollBarVisibility	HmiScrollBarVisibility	Gibt die Sichtbarkeit der vertikalen Bildlaufleiste des Bildfensters an	L/S
VerticalScrollBarPosition	int	Gibt die Position der vertikalen Bildlaufleiste des Bildfensters an	L/S
HorizontalScrollBarVisibility	HmiScrollBarVisibility	Gibt die Sichtbarkeit der horizontalen Bildlaufleiste des Bildfensters an	L/S
HorizontalScrollBarPosition	int	Gibt die Position der horizontalen Bildlaufleiste des Bildfensters an	L/S
Adaption	HmiScreenWindowAdaption	Gibt die Anpassung der Größe des Bildfensters an	L/S
InteractiveZooming	bool	Gibt die interaktive Zoomfunktion des Bildfensters an	L
Caption	MultilingualProperty	Gibt den Titel des Bildfensters an	L/S
WindowFlags	HmiWindowFlag	Gibt die Fensterkonfiguration an, z. B. ShowCaption, ShowBorder, AlwaysOnTop	L/S
Icon	string	Gibt das Symbol des Bildfensters an	L/S
Top	int	Gibt den Wert der Y-Koordinaten des Bildfensters an	L/S
Left	int	Gibt den Wert der X-Koordinaten des Bildfensters an	L/S
Width	uintz	Gibt die Breite des Bildfensters an	L/S
Height	uint	Gibt die Höhe des Bildfensters an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Bildfensters an	L
Name	string	Gibt den Namen des Bildfensters an	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
TabIndex	ushort	Bildobjekte, die einen Tab-Index 0 vorgeben, sind nicht Teil der Tab-Reihenfolge	L/S
EventHandlers	HmiScreenWindowEventHandlerComposition	Gibt die Ereignis-Handler des Bildfensters an	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
CaptionColor	Color	Gibt die Farbe des Titels des Bildfensters an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf Eigenschaften eines Bildfensters zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ScreenWindowPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    var screen = hmiSoftware.Screens;
    var createdscreen = screen.Create("TestScreen_15666");
    var hmiscreenwindow =
    createdscreen.ScreenItems.Create<HmiScreenWindow>("HmiScreenWindow_1")
    // To access basis property of screen window
    //Width
    var autoplay = hmiscreenwindow.Width;
    hmiscreenwindow.Width = 100;
    //HorizontalScrollBarVisibility
    var horizontalscrollbarvisibilityValue = hmiscreenwindow.HorizontalScrollBarVisibility;
    hmiscreenwindow.HorizontalScrollBarVisibility = HmiScrollBarVisibility.Visible;
    //Name
    var name = hmiscreenwindow.Name;
    hmiscreenwindow.Name = "DefaultName";
    // To access multilingual property
    // Caption
    hmiscreenwindow.Caption.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
}
```

Auf Eigenschaften einer Systemdiagnoseanzeige zugreifen (RT Unified)**Eigenschaft**

Die folgenden Eigenschaften werden bei Bildobjekten von Systemdiagnoseanzeigen unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
SystemDiagnosisView	HmiDataGridView-Part	Gibt die Systemdiagnoseansicht an	L
Caption	HmiTextPart	Gibt den Text an, der im Titel des Bildfensters oder Fensterbedienelements (Label) angezeigt werden soll	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
BackColor	Color	Gibt die Hintergrundfarbe der Systemdiagnoseanzeige an	L/S
WindowFlags	HmiWindowFlag	Gibt die Fensterkonfiguration an, z. B. ShowCaption, ShowBorder, AlwaysOnTop	L/S
Icon	string	Gibt das Symbol in der Systemdiagnoseanzeige an	L/S
Top	int32	Gibt den Wert der Y-Koordinaten der Systemdiagnoseanzeige an	L/S
Left	int32	Gibt den Wert der X-Koordinaten der Systemdiagnoseanzeige an	L/S
Width	uint32	Gibt den Breitenwert der Systemdiagnoseanzeige an	L/S
Height	uint32	Gibt den Höhenwert der Systemdiagnoseanzeige an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Systemdiagnoseanzeige an	L
Name	string	Gibt den Namen der Systemdiagnoseanzeige an	L/S
ToolBar	HmiToolBarPart	Gibt die Symbolleiste der Systemdiagnoseanzeige an	L
StatusBar	HmiStatusBarPart	Gibt die Statusleiste in der Systemdiagnoseanzeige an	L
TimeZone	int	Gibt die Zeitzone der Systemdiagnoseanzeige an	L/S
CaptionColor	Color	Gibt die Farbe des Titels der Systemdiagnoseanzeige an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften einer Systemdiagnoseanzeige zuzugreifen, ändern Sie folgenden Programmcode:

```
private void SystemDiagnosisControlPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen screen = hmiSoftware.Screens.Create("Screen_1");
    HmiSystemDiagnosisControl syscontrol =
    screen.ScreenItems.Create<HmiSystemDiagnosisControl>("SysDiag_1");
    syscontrol.SystemDiagnosisView.RowHeight = 50;
    Debug.Assert(syscontrol.SystemDiagnosisView.RowHeight.Equals(50));
    syscontrol.TimeZone = 28;
    Debug.Assert(syscontrol.TimeZone.Equals(28));
}
```

Auf Eigenschaften einer Kurvenanzeige zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Kurvenanzeigen-Bildobjekten unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
ShowStatisticRulers	bool	Gibt an, dass die Statistik-Lineale der Kurvenanzeige eingeblendet werden sollen	L/S
AreaSpacing	ushort	Gibt die Bereichsfläche in der Kurvenanzeige an	L/S
ExtendRulerToAxis	bool	Gibt an, dass das Lineal zusätzlich zur Achse in der Kurvenanzeige eingeblendet werden soll	L/S
Caption	MultilingualProperty	Gibt den Text an, der im Titel des Bildfensters oder Fensterbedienelements (Label) angezeigt werden soll	L/S
WindowFlags	HmiWindowFlag	Gibt die Fensterkonfiguration an, z. B. ShowCaption, ShowBorder, AlwaysOn-Top	L/S
Icon	string	Gibt das Symbol im Steuerfenster an	L/S
Top	int	Gibt den Wert der Y-Koordinaten der Kurvenanzeige an	L/S
Left	int	Gibt den Wert der X-Koordinaten der Kurvenanzeige an	L/S
Width	uint	Gibt den Breitenwert der Kurvenanzeige an	L/S
Height	uint	Gibt den Höhenwert der Kurvenanzeige an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Kurvenanzeige an	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Name	string	Gibt den Namen der Kurvenanzeige an	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
TabIndex	ushort	Bildobjekte, die einen Tab-Index 0 vorgeben, sind nicht Teil der Tab-Reihenfolge	L/S
ToolBar	HmiToolBarPart	Gibt die Symbolleiste der Kurvenanzeige an	L
StatusBar	HmiStatusBarPart	Gibt die Statusleiste in der Kurvenanzeige an	L
Font	HmiFontPart	Gibt die Schriftart der Kurvenanzeige an	L
TimeZone	int	Gibt die Zeitzone in der Kurvenanzeige an	L/S
ShowRuler	bool	Gibt das Lineal der Kurvenanzeige an	L/S
ShiftAxes	bool	Gibt die Verschiebung von Achsen der Kurvenanzeige an	L/S
BackColor	Color	Gibt die Hintergrundfarbe der Kurvenanzeige an	L/S
Legend	HmiLegendPart	Gibt die Legende der Kurvenanzeige an	L
EventHandlers	HmiTrendControlEventsHandlerComposition	Gibt die Ereignis-Handler der Kurvenanzeige an	L
CaptionColor	Color	Gibt die Farbe des Titels der Kurvenanzeige an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Hinweis

LabelFont ist eine Eigenschaft der Zeitachse für Kurvenanzeigen.

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf grundlegende Eigenschaften einer Kurvenanzeige zuzugreifen, ändern Sie folgenden Programmcode:

```
private void TrendControlPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiScreen screen = hmiSoftware.Screens.Create("Screen_164");
    HmiFunctionTrendControl hmitrendcontrol =
    screen.ScreenItems.Create<HmiFunctionTrendControl>("CTrendControl");
    // To access basic properties
    //Width
    var autoplay = hmitrendcontrol.Width;
    hmitrendcontrol.Width = 1;
    //BackColor
    var backcolor = hmitrendcontrol.BackColor;
    hmitrendcontrol.BackColor = Color.Beige;
    //Name
    var name = hmitrendcontrol.Name;
    hmitrendcontrol.Name = "DefaultName";
    // To access multilingual property
    //Caption
    var caption = hmitrendcontrol.Caption.Items[0].Text;
    hmitrendcontrol.Caption.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    //To access other typical properties
    //Status bar
    var statusBar = hmitrendcontrol.StatusBar;
    var backColor = statusBar.BackColor;
    statusBar.BackColor = Color.Aqua;
    var enabled = statusBar.Enabled;
    statusBar.Enabled = true;
    var visible = statusBar.Visible;
    statusBar.Visible = true;
}
```

Auf Eigenschaften einer Wertetabelle zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Wertetabellen-Bildobjekten unterstützt.

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
TrendCompanion-Mode	HmiTrendCompanion-Mode	Gibt die Wertetabelle an	L/S
UseSourceControlTrendColors	bool	Gibt die Farbe der Wertetabelle basierend auf einem Quellen-Bedienelement an	L/S
ShowAlways	bool	Gibt an, ob „Immer anzeigen“ aktiviert ist	L/S
UseSourceControlBackColor	bool	Gibt die Hintergrundfarbe basierend auf einem Quellen-Bedienelement an	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Caption	MultilingualProperty	Gibt den Text an, der im Titel des Bildfens- ters oder Fensterbedienelements (Label) angezeigt werden soll	L/S
WindowFlags	HmiWindowFlag	Gibt die Fensterkonfiguration an, z. B. ShowCaption, ShowBorder, AlwaysOnTop	L/S
Icon	string	Gibt das Symbol im Steuerfenster an	L/S
Top	int	Gibt den Wert der Y-Koordinaten der Wer- tetabelle an	L/S
Left	int	Gibt den Wert der X-Koordinaten der Wer- tetabelle an	L/S
Width	uint	Gibt den Breitenwert der Wertetabelle an	L/S
Height	uint	Gibt den Höhenwert der Wertetabelle an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus der Werteta- belle an	L
Name	string	Gibt den Namen der Wertetabelle an	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ ak- tiviert ist	L/S
TabIndex	ushort	Bildobjekte, die einen Tab-Index 0 vorge- ben, sind nicht Teil der Tab-Reihenfolge	L/S
ToolBar	HmiToolBarPart	Gibt die Symbolleiste der Wertetabelle an	L
StatusBar	HmiStatusBarPart	Gibt die Statusleiste in der Wertetabelle an	L
SourceTrendControl	string	Gibt die Farbe der Kurve basierend auf ei- nem Quellen-Bedienelement an	L/S
TimeZone	int	Gibt die Zeitzone der Wertetabelle an	L/S
SnapToSourceCont- rol	bool	Gibt das Bedienelement an, an dem gefan- gen werden soll	L/S
TrendRulerView	HmiDataGridViewPart	Gibt die Linealansicht der Kurve an	L
TrendStatisticArea- View	HmiDataGridViewPart	Gibt die Statistikbereich-Ansicht an	L
BackColor	Color	Gibt die Hintergrundfarbe der Wertetabel- le an	L/S
TrendStatisticRe- sultView	HmiDataGridViewPart	Gibt die Statistikergebnis-Ansicht an	L
EventHandlers	HmiTrendCompanionE- ventHandlerCompositi- on	Gibt die Ereignis-Handler der Wertetabelle an	L
CaptionColor	Color	Gibt die Farbe des Titels der Wertetabelle an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Ändern Sie folgenden Programmcode, um auf die grundlegenden Eigenschaften einer Wertetabelle zuzugreifen:

```
private void TrendCompanionPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    var screen = hmiSoftware.Screens;
    var createdscreen = screen.Create("TestScreen_1566");
    var hmitrendcompanion =
        createdscreen.ScreenItems.Create<HmiTrendCompanion>("HmiTrendCompanion_1");
    // To access basic properties
    //Width
    var autoplay = hmitrendcompanion.Width;
    hmitrendcompanion.Width = 100;
    //BackColor
    var backcolor = hmitrendcompanion.BackColor;
    hmitrendcompanion.BackColor = Color.Beige;
    //Name
    var name = hmitrendcompanion.Name;
    hmitrendcompanion.Name = "DefaultName";
    // To access multilingual property
    //Caption
    var caption = hmitrendcompanion.Caption.Items[0].Text;
    hmitrendcompanion.Caption.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
    // To access other typical properties
    //Status bar
    var statusBar = hmitrendcompanion.StatusBar;
    var backColor = statusBar.BackColor;
    statusBar.BackColor = Color.Aqua;
    var enabled = statusBar.Enabled;
    statusBar.Enabled = true;
    var visible = statusBar.Visible;
    statusBar.Visible = true;
}
```

Auf Eigenschaften eines Browsers zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei Browser-Bildobjekten unterstützt.

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Url	string	Gibt die URL des Browsers an	L/S
BackColor	Color	Gibt die Hintergrundfarbe des Browsers an	L/S
Caption	MultilingualProperty	Gibt den Text an, der im Titel des Bildfensters oder Fensterbedienelements (Label) angezeigt werden soll	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
WindowFlags	HmiWindowFlag	Gibt die Fensterkonfiguration an, z. B. ShowCaption, ShowBorder, AlwaysOnTop	L/S
Icon	string	Gibt das Symbol im Steuerfenster an	L/S
Top	Int	Gibt den Wert der Y-Koordinaten des Browsers an	L/S
Left	int	Gibt den Wert der X-Koordinaten des Browsers an	L/S
Width	uint	Gibt den Breitenwert des Browsers an	L/S
Height	uint	Gibt den Höhenwert des Browsers an	L/S
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Browsers an	L/S
Name	string	Gibt den Namen des Browsers an	L/S
Visible	bool	Gibt an, ob die Visualisierung aktiviert ist	L/S
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	L/S
TabIndex	ushort	Bildobjekte, die einen Tab-Index 0 vorgeben, sind nicht Teil der Tab-Reihenfolge	L/S
ToolBar	HmiToolBarPart	Gibt die Symbolleiste des Browsers an	L
StatusBar	HmiStatusBarPart	Gibt die Statusleiste in einem Browser an	L
EventHandlers	HmiWebControlEventsHandlerComposition	Gibt den Ereignis-Handler des Browsers an	L
CaptionColor	Color	Gibt die Titelfarbe des Browsers an	L/S
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Ändern Sie folgenden Programmcode, um auf die grundlegenden Eigenschaften eines Browsers zuzugreifen:

```
private void WebControlPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    var screen = hmiSoftware.Screens;
    var createdscreen = screen.Create("TestScreen_1566");
    var hmiwebcontrol = createdscreen.ScreenItems.Create<HmiWebControl>("HmiWebControl_1")
    // To access basic property
    //Width
    var autoplay = hmiwebcontrol.Width;
    hmiwebcontrol.Width = 100;
    //BackColor
    var backcolor = hmiwebcontrol.BackColor;
    hmiwebcontrol.BackColor = Color.Beige;
    //Name
    var name = hmiwebcontrol.Name;
    hmiwebcontrol.Name = "DefaultName";
    // To access multilingual property
    //Caption
    var caption = hmiwebcontrol.Caption.Items[0].Text;
    hmiwebcontrol.Caption.Items[0].Text= "<body><p>TestforMultilingualProperty</p></body>";
    // To access other typical properties
    //Status bar
    var statusBar = hmiwebcontrol.StatusBar;
    var backColor = statusBar.BackColor;
    statusBar.BackColor = Color.Aqua;
    var enabled = statusBar.Enabled;
    statusBar.Enabled = true;
    var visible = statusBar.Visible;
    statusBar.Visible = true;
}
```

Bildbausteincontainer (RT Unified)

Beschreibung Bildbausteincontainer (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um eine Aktivität an Bildobjekten von Bildbausteincontainern durchzuführen, können Sie die Befehle „enumerate“, „find“, „index“ sowie die Methode „Contains()“ verwenden.

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um aus der Bildobjekt-Liste über die Eigenschaft ScreenItems der Klasse „HmiScreen“ auf einen Bildbausteincontainer zuzugreifen, ändern Sie folgenden Programmcode:

```
private void FacePlateContainerSearch(HmiSoftware hmiSoftware)
{
    HmiScreen hmiScreen = hmiSoftware.Screens.Create("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiScreen.Screens[0].ScreenItems;
    foreach (var item in screenitems)
    {
        //working with screenitems
    }
}
```

Um anhand des Namens aus der Bildobjekt-Liste auf einen Bildbausteincontainer zuzugreifen, ändern Sie folgenden Programmcode:

```
private void FacePlateInstanceSearchByName(HmiSoftware hmiSoftware, string
FacePlateContainerName = "FaceplateContainer_1")
{
    HmiScreenItemBase screenitems =
hmiSoftware.Screens[0].ScreenItems.Find(FacePlateContainerName);
}
```

Um anhand des Indexes aus der Bildobjekt-Liste auf einen Bildbausteincontainer zuzugreifen, ändern Sie folgenden Programmcode:

```
private void FaceplateContainerSearchByIndex(HmiSoftware hmiSoftware)
{
    HmiScreenItemBase screenitem = hmiSoftware.Screens[0].ScreenItems[0];
}
```

Um mit Contains() zu ermitteln, ob in einer Bildobjekt-Liste ein bestimmter Bildbausteincontainer vorhanden ist, ändern Sie folgenden Programmcode:

```
private void IsFaceplateContainerExist(HmiSoftware hmiSoftware)
{
    HmiScreenItemBaseComposition screenitems = hmiSoftware.Screens[0].ScreenItems;
    HmiFaceplateContainer faceplate = screenitems.Create<HmiFaceplateContainer>("Faceplate_1");
    bool isexists = screenitems.Contains(faceplate);
}
```

Mit einem Bildbausteincontainer arbeiten

Mit einem Bildbausteincontainer können Sie während der Verwendung von TIA Portal Openness folgende Aktionen durchführen:

- Bildbausteincontainer erstellen: FaceplateContainer.Create() (Seite 824)
- Bildbausteincontainer löschen: FaceplateContainer.Delete() (Seite 824)
- Eigenschaften von Bildbausteincontainern: Auf Eigenschaften von Bildbausteincontainern zugreifen (Seite 825)

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

- Enthaltene Typeigenschaft eines Bildbausteincontainers: Auf enthaltene Typeigenschaften zugreifen (Seite 828)
- UDT-Eigenschaft eines Bildbausteincontainers: UDT einem Bildbausteincontainer zuordnen (Seite 829)

FaceplateContainer.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um ein Bildobjekt für einen Bildbausteincontainer zu erstellen, ändern Sie folgenden Programmcode:

```
private void CreateFaceplateContainer(HmiSoftware hmiSoftware, string
FacePlateContainerName = "FaceplateContainer_1")
{
    HmiScreen hmiScreen= hmiSoftware.Screens.Create(screenName);
    HmiScreenItemBaseComposition screenitems = hmiSoftware.Screens[0].ScreenItems;
    HmiFaceplateContainer faceplate =
    hmiScreen.ScreenItems.Create<HmiFaceplateContainer>(FacePlateContainerName);
}
```

FaceplateContainer.Delete() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um ein Bildobjekt für einen Bildbausteincontainer zu löschen, ändern Sie folgenden Programmcode:

```
private void DeleteFaceplateContainer(HmiSoftware hmiSoftware, string
FaceplateContainerName = "FaceplateContainer_1")
{
    HmiScreen hmiScreen = hmiSoftware.Screens.Create("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiSoftware.Screens[0].ScreenItems;
    HmiFaceplateContainer faceplate =
    hmiScreen.ScreenItems.Create<HmiFaceplateContainer>(FaceplateContainerName);
    If (faceplate != null)
    {
        faceplate.Delete ();
    }
}
```

```
private void DeleteFaceplateContainer(HmiSoftware hmiSoftware, string
FacePlateContainerName = "FaceplateContainer_1")
{
    HmiSoftware hmiSoftware = GetHMISoftware();
    HmiScreen hmiScreen = hmiSoftware.Screens.Create ("Screen_1");
    HmiScreenItemBaseComposition screenitems = hmiSoftware.Screens[0].ScreenItems;
    HmiFaceplateContainer faceplate =
    hmiScreen.ScreenItems.Create<HmiFaceplateContainer>(FacePlateContainerName);
    IEngineeringObject ObjhmiLineEnggObj = faceplate;
    if (ObjhmiLineEnggObj != null)
    {
        ObjhmiLineEnggObj.Invoke ("Delete", null);
    }
}
```

Auf Eigenschaften von Bildbausteincontainern zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei einem Bildbausteincontainer unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugänglichkeit
Name	System.String	Gibt den Namen der Bildbausteininstanz an	R/W
ContainedType	System.String	Gibt den Typ des enthaltenen Bildbausteins an Weitere Informationen zum enthaltenen Bildbausteintyp finden Sie unter Auf enthaltene Typeigenschaften zugreifen (Seite 828)	R/W

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugänglichkeit
Enabled	System.Boolean	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	R/W
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus an	R
Height	System.UInt32	Gibt die Höhe des Steuerfensters an	R/W
Icon	System.String	Gibt das Symbol im Steuerfenster an	R/W
Caption	Siemens.Engineering.MultilingualText	Gibt den Text an, der im Titel des Bildfensters oder Fensterbedienelements (Label) angezeigt werden soll	R
TabIndex	System.UInt16	Gibt die Bildobjekte an, die einen Tab-Index 0 vorgeben und nicht Teil der Tab-Reihenfolge sind	R/W
Visible	System.Boolean	Gibt die Sichtbarkeit eines Bildobjekts an	R/W
Width	System.UInt32	Gibt die Breite eines Steuerfensters an	R/W
WindowFlags	Siemens.Engineering.miUnified.UI.EnumWindowFlag	Gibt die Fensterkonfiguration an, wie ShowCaption, ShowBorder, AlwaysOn-Top.	R/W
Top	System.Int32	Gibt den Wert der Y-Koordinaten des Steuerfensters an	R/W
Left	System.Int32	Gibt den Wert der X-Koordinaten des Steuerfensters an	R/W
Interface	Siemens.Engineering.HmiUnified.UI.Parts.HmiFaceplateInterfaceComposition	Gibt die Bildbaustein-Schnittstellen an	R
EventHandlers	HmiFaceplateContainerEventHandlerComposition	Gibt die Ereignis-Handler von HmiFaceplateContainer an Weitere Informationen über Ereignis-Handler siehe	R
ShowFocusVisual	Boolean	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	R/W
RotationAngle	Int16	Gibt den Drehwinkel des Bildobjekts in Grad an	R/W

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um die Verwendung der Eigenschaften eines Bildbausteincontainers abzurufen, ändern Sie folgenden Programmcode:

```
private void FacePlateContainerPropertiesAccess()
{
    HmiScreen screen = m_hmiSoftware.Screens[0];
    HmiFaceplateContainer faceplate = screens[0].ScreenItems[0] as HmiFaceplateContainer;
    //Get_faceplateContainedType
    Var containedtype = faceplate.ContainedType;
    Console.WriteLine("Faceplate Container Type: " + containedtype);
    Example Result = "Default Version No.\Faceplate type name";
    //Get_CaptionText
    string Caption = faceplate.Caption.Items[0].Text;
    Console.WriteLine("Faceplate Caption: " + Caption);
    //Get_WindowFlags
    HmiWindowFlag WindowFlags = faceplate.WindowFlags;
    Console.WriteLine("Faceplate WindowFlags: " + WindowFlags);
    //Get_Icon
    string Icon = faceplate.Icon;
    Console.WriteLine("Faceplate Icon: " + Icon);
    //Get_Top
    int Top = faceplate.Top;
    Console.WriteLine("Faceplate Top: " + Top);
    //Get_Left
    int Left = faceplate.Left;
    Console.WriteLine("Faceplate Left: " + Left);
    //Get_Width
    uint Width = faceplate.Width;
    Console.WriteLine("Faceplate Width: " + Width);
    //Get_Height
    uint Height = faceplate.Height;
    Console.WriteLine("Faceplate Height: " + Height);
    //Get_CurrentQuality
    HmiQuality CurrentQuality = faceplate.CurrentQuality;
    Console.WriteLine("Faceplate CurrentQuality: " + CurrentQuality);
    //Get_Name
    string Name = faceplate.Name;
    Console.WriteLine("Faceplate Name : " + Name);
    //Get_Visible
    bool Visible = faceplate.Visible;
    Console.WriteLine("Faceplate Visible: " + Visible);
    //Get_Enabled
    bool Enabled = faceplate.Enabled;
    Console.WriteLine("Faceplate Enabled: " + Enabled);
    //Get_TabIndex
    ushort TabIndex = faceplate.TabIndex;
    Console.WriteLine("Faceplate TabIndex: " + TabIndex);
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um die Verwendung der Eigenschaften eines Bildbausteincontainers einzustellen, ändern Sie folgenden Programmcode:

```
private void FacePlateContainerPropertiesAccess()
{
    //Set_ContainedType
    faceplate.ContainedType = "V0.0.1\Faceplate_1";
    //Set_CaptionText:
    faceplate.Caption.Items[0].Text = "testCaption";
    //Set_WindowFlags
    faceplate.WindowFlags = HmiWindowFlag.CanSize;
    //Set_Icon
    faceplate.Icon = "testCaption";
    //Set_Top
    faceplate.Top = 50;
    //Set_Left
    faceplate.Left = 50;
    //Set_Width
    faceplate.Width = 50;
    //Set_Height
    faceplate.Height = 50;
    //Set_Name
    faceplate.Name = "TestName";
    //Set_Visible
    faceplate.Visible = true;
    //Set_Enabled
    faceplate.Enabled = true;
    //Set_TabIndex
    faceplate.TabIndex = 10;
}
```

Auf enthaltene Typeigenschaften zugreifen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die enthaltene Typeigenschaft des Bildbausteincontainers mit neuem Aufbewahrungsort von Bildbausteintypen in der Projektbibliothek zuzugreifen:

```
private void AccessUnifiedFaceplateProperties(createFacePlateContainer)
{
    HmiFaceplateContainer faceplatecontainer = createFacePlateContainer("FacePlateContainer");
    faceplateContainer.ContainedType = "Faceplate_1";
    var containedType = faceplateContainer.ContainedType;
    //Output: "V0.0.2\\Faceplate_1"
    HmiFaceplateContainer faceplatecontainer = CreateFacePlateContainer("FacePlateContainer");
    faceplateContainer.ContainedType = "V0.0.1\\Faceplate_1";
    var containedType = faceplateContainer.ContainedType;
    //Output: "V0.0.1\\Faceplate_1"
}
```

UDT einem Bildbausteincontainer zuordnen (RT Unified)**Voraussetzung**

Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um UDT einem Bildbausteincontainer zuzuordnen, ändern Sie folgenden Programmcode:

```
private void AssignUDTTToFaceplateContainer(HmiSoftware hmiSoftware, Project proj)
{
    var hmiSoftware = GetHmiSoftware(proj);
    HmiScreenItemBaseComposition screenitems =
hmiSoftware.Screens.Find("Screen_1").ScreenItems;
    HmiFaceplateContainer faceplateContainer = screenitems.Find("FaceplateContainer_1") as
HmiFaceplateContainer;
    faceplateContainer.ContainedType = "Faceplate_1";
    // where UDT is the instance created in TagTable from the UDT created in library
    faceplateContainer.Interface[0].Value = "UDT";
}
```

Meine Bedienelemente (RT Unified)

Auf Eigenschaften eines Custom Web Control Containers zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei einem HMI Custom Web Control Container unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Authorization	string	Gibt die Autorisierung des Custom Web Control Containers an	R/W
EventHandlers	HmiCustomWebControlContainerEventHandlerComposition	Gibt die Ereignis-Handler des Custom Web Control Containers an	R
Interface	HmiCustomControlInterfaceComposition	Gibt die Schnittstelle des Custom Web Control Containers an	R
RequireExplicitUnlock	bool	Gibt an, ob eine explizite Entriegelung erforderlich ist	R/W
Left	int	Gibt den Wert der X-Koordinaten des Steuerfensters an	R/W
Caption	HmiTextPart	Gibt den Text an, der im Titel des Bildfensters oder Fensterbedienelements (Label) angezeigt werden soll	R
CaptionColor	Color	Gibt die Titelfarbe des Custom Web Control Containers an	R/W
ContainedType	string	Gibt den Typ des enthaltenen Bildbausteins an	R/W
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Custom Web Control Containers an	R
Dynamization	DynamizationBaseComposition	Gibt die Dynamisierung des Custom Web Control Containers an	R
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	R/W
Height	uint	Gibt die Höhe des Custom Web Control Containers an	R/W
Icon	string	Gibt das Symbol des Custom Web Control Containers an	R/W
Name	string	Gibt den Namen des Custom Web Control Containers an	R/W
PropertyEventHandlers	PropertyEventHandlerComposition	Gibt die Eigenschafts-Ereignis-Handler des Custom Web Control Containers an	R
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	R/W
TabIndex	ushort	Bildobjekte, die einen Tab-Index 0 vorgeben, sind nicht Teil der Tab-Reihenfolge	R/W
Top	int	Gibt den Wert der Y-Koordinaten des Custom Web Control Containers an	R/W

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Visible	bool	Gibt die Sichtbarkeit des Custom Web Control Containers an	R/W
Width	uint	Gibt die Breite des Steuerfensters an	R/W
WindowFlags	HmiWindowFlag	Gibt die Fensterkonfiguration an, z. B. ShowCaption, ShowBorder, AlwaysOn-Top	R/W

Hinweis

Den Pfad für den Zugriff auf den HMI Custom Web Control Container finden Sie unter:
C:\Programme\SiemensAutomation\Portal V*\Data\Hmi\CustomControls

Hierbei steht V* für die installierte Version von TIA Portal.

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf ein Bedienelement „Anlagensicht“ vom Typ „HmiCustomWebControlContainer“ zuzugreifen, ändern Sie folgenden Programmcode:

```
private void CustomWebControlContainerPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware(MyProject);
    HmiScreenComposition screens = hmiSoftware.Screens;
    HmiScreen screen = screens.Create("Screen_1");
    HmiCustomWebControlContainer plantOverview =
    screen.ScreenItems.Create<HmiCustomWebControlContainer>("DefaultName", "Siemens.CPM");
    HmiCustomWebControlContainer reports =
    screen.ScreenItems.Create<HmiCustomWebControlContainer>("DefaultName_1", "Siemens.REP");
    // Access basic properties of plant overview control of type HmiCustom Web Control Container
    //Left
    plantOverview.Left = 1;
    //CaptionColor
    plantOverview.CaptionColor = Color.AliceBlue;
    // Access other typical properties of plant overview control of type Hmi Custom Web Control
    Container
    HmiCustomControlInterfaceComposition customControlInterfaceComposition =
    plantOverview.Interface;
    var count = customControlInterfaceComposition.Count;
    if (count > 0)
    {
        foreach (var customControlInterface in customControlInterfaceComposition)
        {
            string propertyName = customControlInterface.PropertyName;
            object propertyValue = customControlInterface.Value;
            if (propertyName == "SelectionBackColor")
            {
                customControlInterface.Value = Color.Blue;
            }
        }
    }
    ///*-----Alternative Way-----*/
    HmiCustomControlInterface selectionBackColorProperty =
    customControlInterfaceComposition.FirstOrDefault(x => x.PropertyName ==
    "SelectionBackColor");
    if (selectionBackColorProperty != null)
    {
        selectionBackColorProperty.Value = Color.AliceBlue;
    }
}
}
```

Hinweis

Zur Erstellung eines Custom Web Controls muss der in der manifest.json-Datei des Custom Web Controls aufgeführte eindeutige Name als ContainedTypeValue angegeben werden. In diesem Fall wird über die Verwendung des Anzeigenamens kein Custom Web Control-Objekt erstellt. Im obigen Beispiel wird anstelle des Anzeigenamens "Siemens.CPM" verwendet: Anlagensicht.

Dynamische Widgets (RT Unified)

Auf Eigenschaften eines Custom Widget Containers zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden bei einem HMI Custom Widget Container unterstützt:

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Authorization	string	Gibt die Autorisierung des Custom Widget Containers an	R/W
CurrentQuality	HmiQuality	Gibt den Verbindungsstatus des Custom Widget Containers an	R
Dynamizations	DynamizationBaseComposition	Gibt die Dynamisierung des Custom Widget Containers an	R
Enabled	bool	Gibt an, ob „Bedienelement erlauben“ aktiviert ist	R/W
EventHandlers	HmiCustomWidgetContainerEventHandlerComposition	Gibt den Ereignis-Handler des Custom Widget Containers an	R
Height	uint	Gibt die Höhe des Custom Widget Containers an	R/W
Interface	HmiCustomControlInterfaceComposition	Gibt die Schnittstelle des Custom Widget Containers an	R
Left	int	Gibt den Wert der X-Koordinaten des Custom Widget Containers an	R/W
Name	string	Gibt den Namen des Custom Widget Containers an	R/W
Opacity	string	Gibt die Opazität des Custom Widget Containers an	R/W
PropertyEventHandlers	PropertyEventHandlerComposition	Gibt den Eigenschafts-Ereignis-Handler des Custom Widget Containers an	R
RequireExplicitUnlock	bool	Gibt an, ob eine explizite Entriegelung erforderlich ist	R/W
RotationAngle	short	Gibt den Drehwinkel des Custom Widget Containers an	R/W
RotationCenterPlacement	HmiRotationCenterPlacement	Gibt den Drehpunkt des Custom Widget Containers an	R/W
RotationCenterX	float	Gibt den X-Drehpunkt des Custom Widget Containers an	R/W
RotationCenterY	float	Gibt den Y-Drehpunkt des Custom Widget Containers an	R/W
ShowFocusVisual	bool	Gibt an, ob „Blickfeld anzeigen“ aktiviert ist	R/W
TabIndex	ushort	Bildobjekte, die einen Tab-Index 0 vorgeben, sind nicht Teil der Tab-Reihenfolge	R/W
ToolTipText	MultilingualText	Gibt den Tooltip-Text des Custom Widget Containers an	R

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Eigenschaftstyp	Beschreibung	Zugänglichkeit
Top	int	Gibt den Wert der Y-Koordinaten des Custom Widget Containers an	R/W
Visible	bool	Gibt die Sichtbarkeit des Custom Widget Containers an	R/W
VisualizeQuality	bool	Gibt die visuelle Qualität des Custom Widget Containers an	R/W
Width	uint	Gibt die Breite des Custom Widget Containers an	R/W

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf die grundlegenden Eigenschaften von „extended.Flame“ vom Typ „HMI Custom Widget Container“ zuzugreifen, ändern Sie folgenden Programmcode:

```
private void CustomWidgetContainerPropertiesAccess()
{
    HmiSoftware hmiSoftware = GetHmiSoftware(MyProject);
    HmiScreenComposition screens = hmiSoftware.Screens;
    HmiScreen screen = hmiSoftware.Screens.Create("Screen_1");
    HmiCustomWidgetContainer flame =
        screen.ScreenItems.Create<HmiCustomWidgetContainer>("Flame_1", "extended.Flame");
    //Left
    flame.Left = 1;
    //ToolTipText
    flame.ToolTipText.Items[0].Text = "<body><p>TestforMultilingualProperty</p></body>";
}
```

Hinweis

Zur Erstellung des Custom Controls ist der richtige Wert für den Parameter 'containedTypeValue' erforderlich.

Um einen Custom Widget Container über TIA Portal Openness zu erstellen, muss das Schlüsselwort „extended.<Widgetname>“ verwendet werden. Der Grund hierfür ist die Verwendung des in der Manifest-Datei angegebenen Originalnamens anstatt eines Anzeigenamens.

Eigenschafts-Ereignis-Handler (RT Unified)

Beschreibung Eigenschafts-Ereignis-Handler (RT Unified)

Voraussetzung

- Auf das HMI-Unified-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf ein Eigenschafts-Ereignis von Bildobjekten zuzugreifen, ändern Sie folgenden Programmcode:

```
private void AccessPropertyEvent(HmiSoftware hmiSoftware)
{
    HmiScreen screen = ((HmiSoftware)targetSW).Screens.Create("MyScreen");
    HmiCircle hmiCircle = screen.ScreenItems.Create<HmiCircle>("MYCircle");
    var hmiScreenPropeventDyn = screen.PropertyEventHandlers.Create("Enabled",
PropertyEventType.Change);
    var enabledEvent = screen.PropertyEventHandlers.Find("Enabled",
PropertyEventType.Change);
    //Property Event Browse
    Console.WriteLine("CountofEventActions"+ screen PropertyEventHandlers.Count.ToString());
}
```

Mit Eigenschafts-Ereignissen für Bildobjekte arbeiten

Mit Ereignissen für Bilder/Bildobjekte können Sie folgende Aktivitäten durchführen, während Sie TIA Portal Openness verwenden:

- Eigenschafts-Ereignis-Handler erstellen: `PropertyEventHandlers.Create()` (Seite 835)
- Eigenschafts-Ereignis-Handler löschen: `PropertyEventHandlers.Delete()` (Seite 836)
- Eigenschaften von Eigenschafts-Ereignis-Handlern: Auf Eigenschaften von Eigenschafts-Ereignis-Handlern zugreifen (Seite 836)

PropertyEventHandlers.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Unified-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um ein Eigenschafts-Ereignis des Bildobjekt-Ebenenobjekts mit dem PropertyEventHandlers-Navigator zu erstellen, ändern Sie folgenden Programmcode:

```
private void CreatePropertyEventHandler(HmiSoftware hmiSoftware)
{
    HmiScreen screen = ((HmiSoftware)targetSW).Screens.Create("MyScreen");
    HmiCircle hmiCircle = screen.ScreenItems.Create<HmiCircle>("MYCircle");
    //Event Creation
    PropertyEventHandler propertyEvent =
hmiCircle.PropertyEventHandlers.Create("ToolTipText", PropertyEventType.Change);
}
```

PropertyEventHandlers.Delete() (RT Unified)**Voraussetzung**

- Auf das HMI-Unified-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um Eigenschafts-Ereignisse für Bildobjekte zu löschen, ändern Sie folgenden Programmcode:

```
private void DeletePropertyEventHandler()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    PropertyEventHandler propertyEvent =
((HmiSoftware)targetSW).Screens[0].ScreenItems[0].PropertyEventHandlers.Create("ToolTipText",
PropertyEventType.Change);
    var changedEvent = screen.PropertyEventHandlers.Find("ToolTipText",
PropertyEventType.Change);
    //Property Event Deletion
    changedEvent.Delete();
}
```

Auf Eigenschaften von Eigenschafts-Ereignis-Handlern zugreifen (RT Unified)**Voraussetzung**

- Auf das HMI-Unified-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf Eigenschaften eines Eigenschafts-Ereignisses zuzugreifen, ändern Sie folgenden Programmcode:

```
private void AccessPropertyEvent()
{
    //Property Event Get
    PropertyEventType eventType = propertyEvent.EventType;
    string propertyName = propertyEvent.PropertyName;
    IHmiScript hmiScript = propertyEvent.Script;
    //Script Action in Event Get
    bool hmiScriptEventAsync = hmiScript.Async;
    string hmiScriptEventGlobal = hmiScript.GlobalDefinitionAreaScriptCode;
    string hmiScriptEventCode = hmiScript.ScriptCode;
    HmiValidationResult result = hmiScript.SyntaxCheck();
}
```

Über Bildobjekte auf den Querverweisdienst zugreifen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

```
private void AccessCrossReferenceService()
{
    SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
    HmiSoftware hmiSoftware = softwareContainer.Software as HmiSoftware; //Accessing lower
levels until we reach the ScreenComposition.
    ScreenComposition screenComposition = hmiSoftware.Screens;
    Screen screen = screenComposition.Find("Screen_1");
    HmiScreenItemBase hmiScreenControl = screen.ScreenItems.Find("Alarm control_1");
    if (hmiScreenControl != null)
    {
        try
        {
            CrossReferenceService crossReferenceService =
hmiScreenControl.GetService<CrossReferenceService>();
            // .....
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

5.13.1.14 Bildgruppen (RT Unified)

Beschreibung von Bildgruppen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf alle Bildgruppen des Geräts zuzugreifen, ändern Sie folgenden Programmcode:

```
private void ScreenGroupBrowse(HmiSoftware hmiSoftware)
{
    screenGroups = hmiSoftware.ScreenGroups;
    MessageBox.Show(screenGroups.Count.ToString());
    HmiScreenGroupComposition screenGroup2ndLevel = hmiSoftware.ScreenGroups[0].Groups;
    MessageBox.Show(screenGroup2ndLevel.Count.ToString());
}
```

Mit Bildgruppen arbeiten

Mit Bildgruppen können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Bildgruppen erstellen: ScreenGroups.Create() (Seite 838)
- Bildgruppen löschen: ScreenGroups.Delete() (Seite 839)
- Eigenschaften von Bildgruppen: Auf Eigenschaften von Bildgruppen zugreifen (Seite 839)

ScreenGroups.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um mit der Methode Create der Klasse HmiScreenGroupComposition eine Bildgruppe (HmiScreenGroup) zu erstellen, ändern Sie den folgenden Programmcode:

```
private void CreateScreenGroupFirstLevel(HmiSoftware hmiSoftware)
{
    HmiScreenGroupComposition screenGroups = hmiSoftware.ScreenGroups;
    HmiScreenGroup screenGroup1stLevel = hmiSoftware.ScreenGroups.Create("Group_1stLevel");
}
```

ScreenGroups.Delete() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um mit der Methode Delete() der Klasse HmiScreenGroup eine bestimmte Gruppe zu löschen, ändern Sie den folgenden Programmcode:

```
private void ScreenGroupDelete(HmiSoftware hmiSoftware)
{
    HmiScreenGroupComposition screenGroups = hmiSoftware.ScreenGroups;
    HmiScreenGroup screenGroupFindLevelOne = hmiSoftware.ScreenGroups.Find("Group_2");
    screenGroupFindLevelOne.Delete();
}
```

Auf Eigenschaften von Bildgruppen zugreifen (RT Unified)

Eigenschaft

Die folgende Eigenschaften werden in Bildgruppen unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Name	String	Gibt den Bildgruppennamen an	L/S
Groups	HmiScreenGroupComposition	Gibt die Liste der Bildgruppen an	L
Screens	HmiScreenComposition	Gibt die Liste der Bilder an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaft Groups von Bildgruppen zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void AccessScreenGroupProperties(HmiSoftware hmiSoftware)
{
    HmiScreenGroupComposition screenGroups = hmiSoftware.ScreenGroups;
    MessageBox.Show(screenGroups.Count.ToString());
    HmiScreenGroupComposition screenGroup2ndLevel = hmiSoftware.ScreenGroups[0].Groups;
    MessageBox.Show(screenGroup2ndLevel.Count.ToString());
}
```

5.13.1.15 Variablen (RT Unified)**Beschreibung Variablen (RT Unified)****Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um an einer Variablen eine Aktivität auszuführen, können Sie entweder die Variable enumerieren oder Find() verwenden.

```
private void TagBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all the tags of device
    HmiTagComposition hmiTags = hmiSoftware.Tags;
    for each (HmiTag hmiTag in hmiTags)
    {
        //working with tags
    }
}
```

```
private void TagSearch()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all the tags of given tag table
    string tagTableName = "SpecificTagTable";
    HmiTagComposition hmiTags = hmiSoftware.HmiTagTables.Find("tagTableName").Tags;
    for each (HmiTag hmiTag in hmiTags)
    {
        //working with tags
    }
}
```

Um anhand des Namens auf eine einzelne Variable zuzugreifen, ändern Sie den folgenden Code:

```
private void AccessTag()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    //Search for a specific tag
    String hmiTagName = "HMI_Tag_1";
    HmiTag hmiTag = hmiSoftware.HmiTags.Find("hmiTagName");
    if(hmiTag != null)
    {
        // working with tags
    }
}
```

Um anhand des Tabellennamens auf eine Variable zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void AccessTagByName()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for a tag present in given tag table with name "TagTable1"
    HmiTag hmiTag4 = hmiSoftware.HmiTagTables.Find("TagTable1").HmiTags.Find("Tag1");
}
```

Mit Variablen arbeiten

Sie können, während Sie TIA Portal Openness verwenden, die folgenden Aktivitäten an Variablen ausführen:

- Variablen erstellen: Tags.Create() (Seite 842)
- Variablen löschen: Tags.Delete() (Seite 843)
- Variablen-Eigenschaften: Auf Variableneigenschaften zugreifen (Seite 844)
- Querverweisdienst für Variablen: Über Variablen auf den Querverweisdienst zugreifen (Seite 852)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)
HMI-Softwareobjekt (Seite 674)

Tags.Create() (RT Unified)**Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um eine HMI-Variable mit der Methode Create() der Klasse HmiTagComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void CreateTag()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagComposition hmiTagscomp1 = hmiSoftware.HmiTags;
    //Create a tag in the default tag table
    HmiTag hmiTag1 = hmiTagscomp1.Create("Tag1");
    //Create a tag in specific tag table with name "TagTable1"
    //TagTable1 has to exist in your project otherwise it results in recoverable exception
    HmiTag hmiTag2 = hmiTagscomp1.Create("Tag2", "TagTable1");
}
```

Um eine HMI-Variable durch Zugreifen auf die Eigenschaft Tags des TagTable-Objekts zu erstellen, ändern Sie den folgenden Programmcode:

```
private void CreateTag()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagComposition hmiTagcomp2 = hmiSoftware.HmiTagTables.Find("Default tag table").HmiTags;
    //Create a tag in the default tag table.
    HmiTag hmiTag3 = hmiTagcomp2.Create("Tag_3");
    //Create a tag in specific tag table with name "TagTableName"
    //TagTableName has to exist in your project otherwise it results in recoverable exception
    HmiTag hmiTag4 = hmiTagcomp2.Create("Tag_4", "TableTableName");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Tags.Delete() (RT Unified)**Voraussetzung**

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um eine HMI-Variable mit der Methode Delete() der Klasse HmiTag zu löschen, ändern Sie folgenden Programmcode:

```
private void DeleteTag()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    //Delete a tag at HmiSoftware level
    HmiTagComposition hmiTagscomp1 = hmiSoftware.HmiTags;
    HmiTag hmiTag1 = hmiTagscomp1.Find("Tag1");
    hmiTag1.Delete();
}
```

Um eine HMI-Variable durch Zugreifen auf die Eigenschaft Tags des TagTable-Objekts zu löschen, ändern Sie den folgenden Programmcode:

```
private void DeleteTag()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    //Delete a tag with tag name "Tag2" in tag table
    HmiTagComposition hmiTags2 = hmiSoftware.HmiTagTables.Find("Default tag table").HmiTags;
    HmiTag hmiTag2 = hmiTags2.Find("Tag2");
    hmiTag2.Delete();
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Auf Variableneigenschaften zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden von Variablen unterstützt, die auch für Variablen mit UDT (benutzerdefiniertem Datentyp) und ihren Mitgliedern gelten.

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
AccessMode	HmiAccessMode	Gibt die Zugriffsart der Variablen an.	L/S
AcquisitionCycle	String	Gibt das Attribut für Erfassungszyklus der Variablen an.	L/S
AcquisitionMode	HmiTagAcquisitionMode	Gibt die Erfassungsart der Variablen an.	L/S
Comment	MultilingualText	Gibt den abgerufenen und festgelegten Kommentar der Variablen an.	L/S
Connection	String	Gibt die Verbindung der Variablen an.	L/S
DataType	String	Gibt den Datentyp der Variablen an. Sie können Variablen unterschiedliche Datentypen zuordnen, beispielsweise einfache, benutzerdefinierte Datentypen oder Arrays. Wie Sie einer Variablen einen benutzerdefinierten Datentyp zuordnen, erfahren Sie in Einer Variablen einen benutzerdefinierten Datentyp zuweisen (Seite 848)	L/S
MaxLength	UInt32	Gibt die Länge der Variablen an.	L/S
Address	String	Gibt die Adresse der Variablen an.	L/S
InitialMinValue	LowerRange	Gibt den Anfangswert für das Ereignis einer Variablen an.	L
Name	String	Gibt den Namen der Variablen an.	L/S
Persistent	Boolean	Gibt die Persistenz der Variablen an.	L/S
PlcName	String	Gibt den Namen der PLC an.	L
PlcTag	String	Gibt das PLC-Variablenattribut an.	L/S
InitialValue	Object	Gibt den Anfangswert der Variablen an.	L/S
SubstituteValue	HmiSubstituteValue	Gibt den Ersatzwert der Variablen an.	L

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
UpdateId	UInt32	Gibt die aktualisierte ID der Variablen an.	L/S
InitialMaxValue	UpperRange	Gibt die obere Grenze für den Anfangswert der Variablen an.	L
TagTableName	String	Gibt den Namen der Variablen-tabelle an, zu der die Variable gehört.	L
HmiDataType	String	Gibt den HMI-Datentyp der Variablen an.	L/S
LinearScaling	Boolean	Gibt den Wertebereich der HMI-Variablen und die PLC-Variablen an, die einander linear zuzuordnen sind.	L/S
HmiStartValue	Object	Gibt den Anfangswertebereich der HMI-Variablen an, der dem Anfangswert der PLC-Variablen linear zugeordnet werden soll.	L/S
HmiEndValue	Object	Gibt den Endwertebereich der HMI-Variablen an, der dem Endwertebereich der PLC-Variablen linear zugeordnet werden soll.	L/S
PlcStartValue	Object	Gibt den Anfangswertebereich der PLC-Variablen an, der dem Anfangswertebereich der HMI-Variablen linear zugeordnet werden soll.	L/S
PlcEndValue	Object	Gibt den Endwertebereich der PLC-Variablen an, der dem Endwertebereich der HMI-Variablen linear zugeordnet werden soll.	L/S
GmpRelevant	Boolean	Gibt die Änderungen am Variablenwert an, der in einem Audit-Trail protokolliert ist.	L/S

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
TagType	Enum->TagType	Gibt die Art der Variable an, beispielsweise, ob es sich um einen einfachen, benutzerdefinierten Datentyp oder um ein Array handelt.	
Members	HmiTagComposition (HmiTag which is of UDT data type consists of member which itself are Tags.)	Gibt die Sammlung von Mitgliedervariablen von benutzerdefinierten Datentypen oder Elementvariablen von Arrays an. Wie Sie auf Eigenschaften von Mitgliedervariablen von benutzerdefinierten Datentypen zugreifen, erfahren Sie in der Anleitung in Auf Eigenschaften von Variablenmitgliedern von benutzerdefinierten Typen zugreifen (Seite 850)	

Die folgenden Eigenschaften sind im unteren/oberen Bereich verfügbar:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Value	Object	Gibt den unteren und oberen Wert der Variablen an.	L/S
ValueType	HmiLimitValueType	Gibt den abgerufenen und festgelegten Werttyp der Variablen an.	L/S

Die folgenden Eigenschaften sind in einem Ersatzwert verfügbar:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Value	Object	Gibt den abgerufenen und festgelegten Werttyp an.	L/S
SubstituteValueUsage	Hmi SubstituteValueUsage	Gibt an, wann der Ersatzwert verwendet wird.	L/S

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Eigenschaften der Variablen unabhängig vom Datentyp zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void TagProperties()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    //Tag_1 has to exist in TIA device project.
    HmiTag hmiTag = hmiSoftware.HmiTags.Find("Tag_1");
    //Enumeration type properties
    AccessMode accessMode = hmiTag.AccessMode;
    UpdateScope scope = hmiTag.UpdateScope;
    //Assign valid value to properties which are independent from datatype of the tag
    hmiTag.Name = "Tag_2";
    hmiTag.AcquisitionCycle = "T1s";
    hmiTag.DataType = "int";
    hmiTag.PlcTag = "PlcTag_1";
}
```

Um auf die Eigenschaften der Variablen abhängig vom Datentyp zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void TagProperties()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    //Tag_1 has to exist in TIA device project.
    HmiTag hmiTag = hmiSoftware.HmiTags.Find("Tag_1");
    //Enumeration type properties
    AccessMode accessMode = hmiTag.AccessMode;
    UpdateScope scope = hmiTag.UpdateScope;
    //Substitute value depends on data type of tag.
    hmiTag.SubstituteValue.Value = "144";
    hmiTag.SubstituteValue.OnCommunicationError = false;
    //Start value depends on data type of tag.
    hmiTag.StartValue = "1";
    hmiTag.LowerLimit.ValueType = LimitValueType.Constant;
    hmiTag.LowerLimit.Value = "HmiTag_1";
}
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Um eine Zeichenkette für einen mehrsprachigen Text zuzuweisen und in jeder Sprache des Projekts eine Übersetzung hinzuzufügen, ändern Sie den folgenden Programmcode:

```
private void TagProperties()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    //Tag_1 has to exist in TIA device project.
    HmiTag hmiTag = hmiSoftware.HmiTags.Find("Tag_1");
    //Enumeration type properties
    AccessMode accessMode = hmiTag.AccessMode;
    UpdateScope scope = hmiTag.UpdateScope;
    //Comment property (multilingual text)
    string culture = "en-US";
    //get the language based on given culture. Culture is standard and it should have correct
    value.
    Language language = GetDeviceProject().LanguageSettings.Languages.Find(new
    System.Globalization.CultureInfo(culture));
    //get comment as multilingual text.
    MultilingualText multiLingualComment = hmiTag.Comment;
    //get all text items from comment property.
    MultilingualTextItemComposition texts = multiLingualComment.Items;
    //find text from text list based on language.
    MultilingualTextItem textItemEnglish = texts.Find(language);
    //get value in selected culture.
    string commentEng = textItemEnglish.Text;
}
```

Siehe auch

- Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen (Seite 131)

Einer Variablen einen benutzerdefinierten Datentyp zuweisen (RT Unified)

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um einer Variablen ohne Verbindung einen benutzerdefinierten Datentyp zuzuweisen, ändern Sie den folgenden Programmcode:

```
private void AssignUdtWithoutConnection()
{
    //Assigning three types of UDT to tag by calling DataType property of tag
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagComposition tags = hmiSoftware.HmiTags;
    tags[0].DataType = @"Project library\Types\HmiUdt_Int\V 0.0.1";
    tags[1].DataType = @"Project library\Types\HmiUdt_byte\V 0.0.1";
    tags[2].DataType = @"Project library\Types\New folder_2\HmiUdt_string\V
0.0.1";
    tags[3].DataType = @"Project library\Types\New folder_1\HmiUdt_string\V
0.0.1";
    tags[4].DataType = @"Project library\Types\New folder_3\HmiUdt_string\V
0.0.1";
}
```

Um einer Variablen mit Verbindung einen benutzerdefinierten Datentyp zuzuweisen, ändern Sie den folgenden Programmcode:

```
private void AssignUdtWithConnection()
{
    //Assigning user defined datatype to tag by calling DataType property of
    tag with connection
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagComposition tags = hmiSoftware.HmiTags;
    tags[0].Connection = "Connection_1";
    tags[0].DataType = @"Project library\Types\New folder_3\HmiUdt_string\V
0.0.1";
}
```

Hinweis

Um einer Variablen mit Verbindung einen benutzerdefinierten Datentyp zuzuweisen, muss im TIA Portal-Projekt die richtige PLC-Verbindung vorhanden sein. Andernfalls wird eine wiederherstellbare Ausnahme zurückgegeben.

Um einer Variablen mit Verbindung einen benutzerdefinierten PLC-Datentyp zuzuweisen, ändern Sie den folgenden Programmcode:

```
private void AssignPlcUdtWithConnection()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagComposition tags = hmiSoftware.HmiTags;
    tags[0].Connection = "HMI_Connection_1";
    tags[0].Data Type = "PlcTag_1";
}
```

Hinweis

Um einer Variablen einen benutzerdefinierten PLC-Datentyp zuzuweisen, müssen sowohl die PLC als auch das HMI-Gerät im selben TIA Portal vorhanden sein und es muss eine integrierte Verbindung zwischen ihnen bestehen.

Die PLC benötigt außerdem eine PLC-Variable mit UDT.

Siehe auch

Projekt öffnen (Seite 131)

Auf Eigenschaften von Variablenmitgliedern von benutzerdefinierten Typen zugreifen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um auf die Mitgliedereigenschaften benutzerdefinierter Datentypen für Variablen zuzugreifen, ändern Sie den folgenden Programmcode:

```
private static void GetProperties(HmiTagComposition hmiTags)
{
    foreach (HmiTag hmiTag in hmiTags)
    {
        switch(hmiTag.TagType)
        {
            case HmiTagType.Simple
            Console.WriteLine("Name : "+ hmiTag.Name);
            GetSimpleTagProperties(HmiTag hmiTag)
            break;
            case HmiTagType.UDT;
            case HmiTagType.Array;
            GetProperties(hmiTag.Members); //recursive call to current function.
            break;
        }
    }
}
//The GetSimpleTagProperties()uses hmiTag object as a parameter to access the properties of
tag.
private static void GetSimpleTagProperties(HmiTag hmiTag)
{
    Console.WriteLine("Name : " +hmiTag.Name);
    Console.WriteLine("Date type : " +hmiTag.DataType);
    // List the properties of hmi tag
    //
    //...
}
//The GetProperties()uses hmiTag.Members object as a parameter to access the properties of
tag members.
private static void GetProperties(HmiTag hmiTag.Members)
{
    Console.Writeline("Name : " +hmiTag.Members.Name);
    Console.Writeline("Date type: " +hmiTag.Members.DataType);
    // List the properties of hmi tag members
    //...
    //...
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Über Variablen auf den Querverweisdienst zugreifen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Ändern Sie den folgenden Programmcode, um auf den Querverweisdienst über GetService() auf dem Variablenobjekt zuzugreifen:

```
private void getCrossReferenceServiceforTag()
{
    SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
    HmiSoftware hmiSoftware = softwareContainer.Software as HmiSoftware; //Accessing lower
    levels until we reach the HmiTagComposition.
    HmiTagComposition hmiTagComposition = hmiSoftware.HmiTags;
    HmiTag hmiTag = hmiTagComposition.Find("Tag_1");
    if (hmiTag != null)
    {
        try
        {
            CrossReferenceService crossReferenceService = hmiTag.GetService<CrossReferenceService>();
            .....
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

HMI-Softwareobjekt (Seite 674)

5.13.1.16 TagTables (RT Unified)

Beschreibung TagTables (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um an einer Variablentabelle eine Aktivität auszuführen, können Sie entweder die Variablentabellen enumerieren oder Find() verwenden.

```
private void TagTableBrowse()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // User can navigate all the tag tables of device
    HmiTagTableComposition tagTables = hmiSoftware.TagTables;
    foreach (HmiTagTable tagTable in tagTables)
    {
        //Working with tag table
    }
}
```

Um anhand des Namens auf eine einzelne Variablentabelle zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void TagTableSearch()
{
    HmiSoftware hmiSoftware = GetHmiSoftware();
    // Search for a specific tag table name from a list of tag tables
    string tagTableName = "TagTable_1"
    HmiTagTableComposition tagTables = hmiSoftware.TagTables;
    HmiTagTable tagTableObj = tagTables.Find("tagTableName");
    //Working with tag table
}
```

Mit Variablentabellen arbeiten

Mit Variablentabellen können Sie folgende Aktionen durchführen, während Sie TIA Portal Openness verwenden:

- Variablentabellen erstellen: TagTables.Create() (Seite 854)
- Variablentabellen löschen: TagTables.Delete() (Seite 854)
- Variablen exportieren: TagTable.Tags.Export() (Seite 861)
- Variablen importieren: TagTable.Tags.Export() (Seite 861)
- Eigenschaften von Variablentabellen: Auf Eigenschaften von Variablentabellen zugreifen (Seite 855)

Hinweis

Die Abfrage auf Fehler wird für Variablentabellen nicht unterstützt, da sie keine Eigenschaften haben, die im Fehlerzustand sein können.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

TagTables.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Variablentabelle mit der Methode Create() der Klasse HmiTagTableComposition zu erstellen, ändern Sie folgenden Programmcode:

```
private void TagTableCreate()
{
    //Create tag table with name "TagTable_1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagTableComposition tagTables = hmiSoftware.TagTables;
    HmiTagTable hmiTagTable = tagTables.Create("TagTable_1");
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

TagTables.Delete() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Variablentabelle mit der Methode Delete() der Klasse HmiTagTable zu löschen, ändern Sie folgenden Programmcode:

```
private void TagTableDelete()
{
    //Delete tag table with name "TagTable_1"
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagTableComposition tagTables = hmiSoftware.TagTables;
    HmiTagTable tagTable = tagTables.Find("TagTable_1");
    tagTable.Delete();
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Auf Eigenschaften von Variablentabellen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in Variablentabellen unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Name	String	Gibt den Namen der Variablentabelle an.	R/W
Tags	HmiTagComposition	Gibt die Liste der Variablen an.	R

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um auf die Eigenschaften einer Variablentabelle zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void TagTableProperties()
{
    //Set and Get the "Name" properties of tag table.
    HmiSoftware hmiSoftware = GetHmiSoftware();
    HmiTagTableComposition tagTables = hmiSoftware.TagTables;
    HmiTagTable tagTable = tagTables.Find("TagTable_1");
    tagTable.Name = "NewTagTable";
    string name = tagTable.Name;
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Variablentabellen exportieren und importieren (RT Unified)

Einleitung

Sie können mit TIA Portal Openness Daten für eine Variablentabelle exportieren und importieren. Wenn nötig, werden beim Import unter dem Unified-Gerät neue Objekte erstellt. Die Exportdatei ist eine Textdatei im vereinfachten Format: WinCC ML, das auf dem YAML-Format basiert. Sie können die Datei entsprechend Ihren Anforderungen ändern.

In TIA Portal Openness sind die Export- und Importmethode auf der Ebene der Variablentabelle implementiert.

Hinweis

Es empfiehlt sich, alle Übersetzungsfehler im Zusammenhang mit zu exportierenden Variablen in Variablentabellen zu beheben.

Variablen exportieren

Mit der Exportfunktion können die folgenden Variablentypen exportiert werden:

- Variablen
 - Interne Variablen
 - Externe Variablen
- Strukturvariablen mit ihren Elementen
 - HMI-Anwenderdatentypen (UDT)
 - Benutzerdefinierte PLC-Datentypen (UDT)
- Array-Variablen mit ihren Elementen
 - Interne Array-Variablen
 - Externe Array-Variablen

Hinweis

Wenn ein Array des Datentyps WChar oder Char exportiert wird, werden in der exportierten WinCCML-Datei keine Variablenelemente angezeigt. Die Variablen werden in der einfachen Variablenkategorie angezeigt. Wenn diese Variablen zurückimportiert werden, werden Sie wie erwartet als Array-Variablen hinzugefügt.

Hinweis

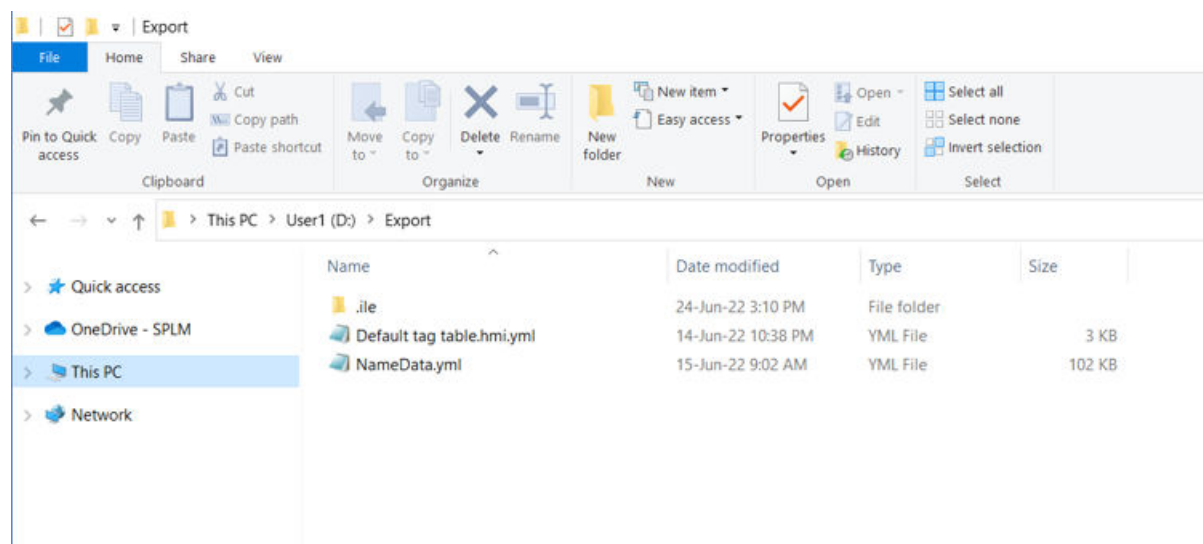
Beim Export einer Strukturvariablen wird der Referenzname des PLC-Strukturvariablentyps an den PLC-Namen angehängt.

Inhalt des Export-Speicherorts

Wenn eine Variablentabelle an einen bestimmten Speicherort im Ordner exportiert wird, werden die folgenden Inhalte erstellt:

- Variablentabellendatei mit der Erweiterung ".hmi.yml" und benutzerdefiniertem Namen im Beispiel "Default tag table.hmi.yml".
- Eine Datei "NameData.yml" – Diese Datei enthält keine Daten, die zur Ausführung des Imports und Exports erforderlich sind.
- Einen Ordner ".ile" – Dieser Ordner enthält keine Daten, die zur Ausführung des Imports und Exports erforderlich sind.
- Eine Datei mit der Erweiterung "def.hmi.yml" wird erstellt, wenn eine Variablentabelle mit Strukturvariablen exportiert wird. Diese Datei enthält die Definition der Strukturtypen.

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

**Struktur der YAML-Datei**

Die Struktur der YAML-Datei ist nachfolgend dargestellt. Sie kann in jedem einfachen Texteditor wie Notepad, Microsoft Word, Notepad++ usw. geöffnet und bearbeitet werden:

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Die gesamte Konfiguration wird als Eigenschaftsname und Eigenschaftswert angezeigt, wodurch die Konfiguration leicht zu lesen und zu ändern ist.

```
#Version: 2.0
SimpleTags:
Tag_1:
AccessMode: SymbolicAccess
AcquisitionCycle: T1s
Address: 0001:TS:50.73A8C75D.C
Connection: HMI_Connection_2
ControllerDataType: 1006
DataType: Int
InitialMaxValue: 444
InitialMinValue: 44
MaxLength: 2
PlcTag: PLCTag_1
Tag_2:
AccessMode: AbsoluteAccess
AcquisitionCycle: T1s
Address: 0001:CL:%IW0
Connection: HMI_Connection_1
ControllerDataType: 1006
DataType: Int
MaxLength: 2
PlcTag: PLCTag
SymbolicAddress: '%IW0'
Tag_3:
AccessMode: SymbolicAccess
AcquisitionCycle: T1s
Address: $0001:TS:8A0E0001.CDB29A10.A.#{0}
Connection: HMI_Connection_1
ControllerDataType: 1009
DataType: DInt
MaxLength: 4
PlcTag: Data_block_1.Element1[0]
SymbolicAddress: Data_block_1.Element1[Tag_1]
Tag_4:
AccessMode: SymbolicAccess
AcquisitionCycle: T1s
Address: 0001:TS:8A0E0001.CDB29A10.A.1
Connection: HMI_Connection_1
ControllerDataType: 1009
DataType: DInt
MaxLength: 4
PlcTag: Data_block_1.Element1[1]
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Eigenschaft

Die folgende Tabelle zeigt die Entsprechungen zwischen den Eigenschaftsnamen in TIA Portal und den Namen der exportierten Eigenschaften in der WinCCML-Datei:

Name der Eigenschaft im TIA Portal	Name der Eigenschaft in WinCCML	Bemerkung
Name	Header des Objekts	
DataType	ControllerDataType	Bei anderen Verbindungen als internen Verbindungen wird diese Eigenschaft beim Import verwendet, um der Variablen einen Datentyp zuzuweisen.
Verbindung	Connection	
PLC-Name	Aus PLC-Variableneigenschaft abgeleitet	
PLC-Variable	PlcTag	
Adresse	SymbolicAddress	Die Eigenschaft 'Adresse' wird vom TIA Portal ES beim Import ignoriert
Zugriffsart	AccessMode	
HMI-Datentyp	DataType	Bei internen Verbindungsvariablen wird diese Eigenschaft beim Import verwendet, um der Variablen einen Datentyp zuzuweisen.
Länge	MaxLength	
Erfassungszyklus	AcquisitionCycle	
Erfassungsart	AcquisitionMode	
Oben 2 (Variable)	MaxValueTag	
Unten 2 (Variable)	MinValueTag	
Oben 2 (Konstante)	InitialMaxValue	
Unten 2 (Konstante)	InitialMinValue	
Lineare Skalierung	Abgeleitet von Anfangs-/Endwerten für PLC/HMI	Wird auf Wahr gesetzt, wenn Anfangs-/Endwerte für PLC/HMI vorhanden sind.
Endwert PLC	PLCMaxValue	
Anfangswert PLC	PLCMinValue	
Endwert HMI	HMIMaxValue	
Anfangswert HMI	HMIMinValue	
Anfangswert	InitialValue	
Aktualisierungskennung	ID	
Persistent	Persistent	
Ersatzwert	SubstituteValue	
Ersatzwert verwenden	SubstituteValueUsage	
GMP-relevant	GMPRelevant	
Art der Bestätigung	ConfirmationType	
Kommentar erforderlich	Abgeleitet aus dem Wert für ConfirmationType	
Gültigkeitsbereich	Scope	
Kommentar	Comment	
	Adresse	

Name der Eigenschaft im TIA Portal	Name der Eigenschaft in WinCCML	Bemerkung
	Offset	
	Slope	

TagTable.Tags.Export() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Variablentabelle zu exportieren, ändern Sie den folgenden Programmcode:

```
DirectoryInfo directoryInfo = new DirectoryInfo("D:\\Data Exchange");
HmiTagTableComposition tables = GetHmiSoftware(currentProject).TagTables;
HmiTagTable tagTable = tables.Find(tagTableName.Text);
\\Create a file name "ExportedTable" in specific directory by using Export() that includes
the details of tag table.
IList<FileInfo> Exportresult =
tagTable.Tags.Export(directoryInfo, "ExportedTable").ToList();
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

TagTable.Tags.Export() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um eine Variablentabelle zu importieren, ändern Sie den folgenden Programmcode:

```
DirectoryInfo directoryInfo = new DirectoryInfo("D:\\Data Exchange");
HmiSoftware ImportHMISoftware = GetHmiSoftware(currentProject);
HmiTagTableComposition tables = ImportHMISoftware.TagTables;
HmiTagTable tagTable = tables.Find(importedTagTable.Text);
bool ImportResult = tagTable.Tags.Import(directoryInfo);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.1.17 Variablentabellengruppen (RT Unified)

Beschreibung von Variablentabellengruppen (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Ändern Sie den folgenden Programmcode, um mit der Methode Find() auf eine Variablentabellengruppe zuzugreifen:

```
privat void SearchTagTableGroups(HmiSoftware hmiSoftware)
{
    HmiTagTableGroupComposition tagTableGroups = hmiSoftware.TagTableGroups;
    HmiTagTableGroup tagTableGroupFindLevelOne = hmiSoftware.tagTableGroups.Find("Group_2");
    MessageBox.Show("Found at 1st level: " + tagTableGroupFindLevelOne.Name);
}
```

Mit Variablentabellengruppen arbeiten

Mit Variablengruppen können Sie während der Verwendung von TIA Portal Openness folgende Aktionen durchführen:

- Variablentabellengruppen erstellen: TagTableGroups.Create() (Seite 862)
- Variablentabellengruppen löschen: TagTableGroups.Delete() (Seite 863)
- Eigenschaften von Variablentabellengruppen: Auf Eigenschaften von Variablentabellengruppen zugreifen (Seite 863)

TagTableGroups.Create() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMISoftware (Seite 676)

Programmcode

Um mit der Methode Create() der Klasse HmiTagTableGroupComposition eine Variablentabellengruppe (HmiTagTableGroup) zu erstellen, ändern Sie den folgenden Programmcode:

```
private void CraeteTagTableGroup(HmiSoftware hmiSoftware)
{
    HmiTagTableGroupComposition tagTableGroups = hmiSoftware.TagTableGroups;
    int count = tagTableGroups.Count;
    HmiTagTableGroup tagTableGroup1stLevel =
hmiSoftware.TagTableGroups.Create("Group_1stLevel");
}
```

TagTableGroups.Delete() (RT Unified)

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HmiSoftware (Seite 676)

Programmcode

Um eine Variablentabellengruppe zu löschen, ändern Sie den folgenden Programmcode:

```
private void DeleteTagTableGroup(HmiSoftware hmiSoftware)
{
    HmiTagTableGroupComposition tagTableGroups = hmiSoftware.TagTableGroups;
    HmiTagTableGroup tagTableGroupFindLevelOne = hmiSoftware.TagTableGroups.Find("Group_2");
    MessageBox.Show("Found at 1st level: " + tagTableGroupFindLevelOne.Name);
    tagTableGroupFindLevelOne.Delete();
}
```

Auf Eigenschaften von Variablentabellengruppen zugreifen (RT Unified)

Eigenschaft

Die folgenden Eigenschaften werden in Variablentabellengruppen unterstützt:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Name	String	Gibt den Variablentabellen-gruppennamen an	L/S
Groups	HmiTagTableGroupComposi-tion	Gibt die Liste der Variablenta-bellengruppen an	L
TagTables	HmiTagTableComposition	Gibt die Liste der Variablenta-bellen an	L

Voraussetzung

- Auf das HMI-Softwareobjekt kann zugegriffen werden.
Siehe Beschreibung HMI Software (Seite 676)

Programmcode

Um auf die Eigenschaft Groups von Variablentabellengruppen eines Geräts zuzugreifen, ändern Sie den folgenden Programmcode:

```
private void AccessTagTableGroupProperties(HmiSoftware hmiSoftware)
{
    HmiTagTableGroupComposition tagTableGroups = hmiSoftware.TagTableGroups;
    MessageBox.Show(tagTableGroups.Count.ToString());
    HmiTagTableGroupComposition tagTableGroup2ndLevel = hmiSoftware.TagTableGroups[0].Groups;
    MessageBox.Show(tagTableGroup2ndLevel.Count.ToString());
}
```

5.13.2 Auf weitere Objekte zugreifen (RT Unified)**5.13.2.1 Umbenennen von Eigenschaften und Datentypen (RT Unified)****Einleitung**

In ES TIA Portal Openness werden Änderungen von Eigenschaftennamen und Datentypen vorgenommen, sodass es dem Runtime Openness-Modell angepasst wird.

Hierfür werden Änderungen in allen Teilsystemen wie zum Beispiel Meldungen, Variablen, Verbindung und Protokollierung vorgenommen.

Für eine einheitliche Benennung der Eigenschaft und des Datentyps muss das Präfix Hmi für den Typ verwendet werden und im Namen der Eigenschaft sollte dieses Präfix nicht erscheinen.

Änderung in Eigenschaftensname und Datentyp

Die folgende Tabelle zeigt die erforderlichen Änderungen für den Erhalt einheitlicher Namen und Datentypen für Zusammensetzungen.

HMI Objektname	Name der Eigenschaft	Neuer Name der Eigenschaft	Datentyp der Eigenschaft	Neuer Datentyp der Eigenschaft
Analogmeldung	AnalogAlarms	-	AnalogAlarmComposition	HmiAnalogAlarmComposition
Bitmeldung	DiscreteAlarms	-	DiscreteAlarmComposition	HmiDiscreteAlarmComposition
OPC UA-Meldungen & Zustandstypen	OpcUaAlarmTypes	-	OpcUaAlarmTypeComposition	HmiOpcUaAlarmTypeComposition

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

HMI Objektname	Name der Eigenschaft	Neuer Name der Eigenschaft	Datentyp der Eigenschaft	Neuer Datentyp der Eigenschaft
Meldeklasse	AlarmClasses	-	AlarmClassComposition	HmiAlarmClassComposition
Variablentabelle	HmiTagTables	TagTables	HmiTagTableComposition	-
Variable	HmiTags	Tags	HmiTagComposition	-
Systemvariable	HmiSystemTags	SystemTags	HmiSystemTagComposition	-
Archivvariable	LoggingTags	-	LoggingTagComposition	HmiLoggingTagComposition
Verbindung	HmiConnections	Connections	HmiConnectionComposition	-
Datenarchiv	DataLogs	-	DataLogComposition	HmiDataLogComposition
Meldearchiv	AlarmLogs	-	AlarmLogComposition	HmiAlarmLogComposition
Schnittstelle Anlagenobjekt	PlantObjectTags	-	PlantObjectInterfaceComposition	-
Bild	Screens	-	HmiScreenComposition	-

Hinweis

Das Zeichen "-" in den Spalten 'Neuer Name' und 'Neuer Datentyp' in obiger Tabelle zeigt an, dass keine Änderung erforderlich ist.

Änderungen in Namen der HMI-Objektklassen

Die folgende Tabelle zeigt die erforderlichen Änderungen für den Erhalt einheitlicher Namen der HMI-Objektklassen.

HMI Objektname	Klassenname	Neuer Klassenname
Analogmeldung	AnalogAlarm	HmiAnalogAlarm
Bitmeldung	DiscreteAlarm	HmiDiscreteAlarm
OPC UA-Meldungen & Zustandstypen	HmiOpcUaAlarmType	HmiOpcUaAlarmType
Meldeklasse	AlarmClass	HmiAlarmClass
Variablentabelle	HmiTagTable	-
Variable	HmiTag	-
Verbindung	HmiConnection	-
Laufzeiteinstellungen	RuntimeSetting	HmiRuntimeSetting
Datenarchiv	DataLog	HmiDataLog
Meldearchiv	AlarmLog	HmiAlarmLog
Archivvariable	LoggingTag	HmiLoggingTag
Systemvariable	HmiSystemTag	-
Bild	HmiScreen	-
Schnittstelle Anlagenobjekt	PlantObjectInterface	-

Hinweis

Das Zeichen "-" in der Spalte 'Neuer Klassenname' in obiger Tabelle zeigt an, dass keine Änderung erforderlich ist.

5.13.2.2 Fehler abfragen (RT Unified)

Einleitung

Mit TIA Portal Openness können Sie Fehlerinformationen über Eigenschaften für folgende WinCC-Objekte abfragen:

- Analogmeldung
- Bitmeldung
- Meldeklasse
- Datenarchiv
- Meldearchiv
- Variable
- Verbindung
- Archivvariable
- Laufzeiteinstellungen von Geräten
- OPC UA-Meldungen

Hinweis

Die Funktion ist nicht auf Systemvariablen und Variablentabellen anwendbar, denn diese haben keine Eigenschaft, die im Fehlerzustand sein kann.

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131).
- Zugriff auf HMI-Unified-Software-Objekt
Siehe HMI-Unified-Software-Objekt (Seite 676)

Programmcode

Ändern Sie folgenden Programmcode, um Fehler zu Variablen, Meldungen, Laufzeiteinstellungen und Bildern abzufragen:

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

```
//For Tag
HmiSoftware hmiSoftware = GetHmiSoftware();
HmiTagComposition hmiTags = hmiSoftware.HmiTags;
HmiTag hmiTag = hmiTags.Find("Tag-1");
if(hmiTag != null)
{
//validate method usage
IList<HmiValidationResult> validationResult = hmiTag.Validate();
if (validationResult == null && validationResult.Count > 0)
{
foreach (HmiValidationResult propertyErrors in validationResult)
{
//browse error for different property
string propertyName = propertyErrors.PropertyName;
foreach (string propertyError in propertyErrors.Errors)
{
//work with propertyerrors
}
}
}
}
//For Discrete Alarm
HmiSoftware hmiSoftware = ...;
HmiDiscreteAlarmComposition discreteAlarms = hmiSoftware.DiscreteAlarms;
HmiDiscreteAlarm alarm = discreteAlarms.Find("Alarm_1");
if (alarm != null)
{
//Validate Method Usage
IList<HmiValidationResult> validationResult = alarm.Validate();
if (validationResult != null && validationResult.Count > 0)
{
foreach (HmiValidationResult propertyErrors in validationResult)
{
//browse error for different property
string propertyName = propertyErrors.PropertyName;
foreach (string propertyError in propertyErrors.Errors)
{
//work with property errors
}
}
}
}
//For Runtime Setting
HmiSoftware hmiSoftware = ...;
HmiRuntimeSetting runtimeSettings = hmiSoftware.RuntimeSettings;
//Validate Method Usage
IList<HmiValidationResult> validationResult = runtimeSettings.Validate();
if (validationResult != null && validationResult.Count > 0)
{
foreach (HmiValidationResult propertyErrors in validationResult)
{
//browse error for different property
string propertyName = propertyErrors.PropertyName;
foreach (string propertyError in propertyErrors.Errors)
{
}
```


5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

```

//work with property errors
}
}
}
// List of Many objects (Tag, Alarm, etc,)
HmiSoftware hmiSoftware = ...;
IList<IValidator> objectsToValidate = new List<IValidator>();
//Put tags in validation list
HmiTagComposition hmiTags = hmiSoftware.Tags;
foreach (HmiTag hmiTag in hmiTags)
{
objectsToValidate.Add(hmiTag);
}
//Put alarms in validation list
HmiDiscreteAlarmComposition discreteAlarms = hmiSoftware.DiscreteAlarms;
foreach (HmiDiscreteAlarm discreteAlarm in discreteAlarms)
{
objectsToValidate.Add(discreteAlarm);
}
//Put RuntimeSettings in validation list
objectsToValidate.Add(hmiSoftware.RuntimeSettings);
foreach (IValidator validator in objectsToValidate)
Ilist<HmiValidationResult> validationResult = validator.Validate();
if (validationResult != null && validationResult.Count > 0)
{
foreach (HmiValidationResult propertyErrors in validationResult)
{
//browse error for different property
string propertyName = propertyErrors.PropertyName;
foreach (string propertyError in propertyErrors.Errors)
{
//work with property errors
}
}
}
}

```

Siehe auch

Beschreibung HMI Software (Seite 676)

5.13.2.3 Auf einfache Dynamisierung zugreifen (RT Unified)**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Es besteht Zugriff auf das HMI-Softwareobjekt
Siehe HMI-Unified-Software-Objekt (Seite 676)

Einleitung

Sie können mit TIA Portal Openness über einfache Dynamisierung vollständig auf die Dynamisierung zugreifen. Das Bild und die Bildelemente können mithilfe einer der folgenden Optionen zur einfachen Dynamisierung dynamisiert werden:

- Bereich
- Mehrere Bits
- Einzelbit

Programmcode

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

```

var hmiSoftware = GetScadaHmiSoftware();
var Screen1 = hmiSoftware.Screens[0];
//Range :
var screenItem0 = Screen1.ScreenItems.FirstOrDefault(x => x.Name == "Circle_1");
string propertyName = "BackColor";
var tagDynamization = screenItem0.Dynamizations.Where(x => x.PropertyName ==
propertyName).FirstOrDefault() as TagDynamization;
var mappingTable = tagDynamization.ValueConverter.MappingTable;
var entries = mappingTable.Entries;
mappingTable.ConditionType = ConditionType.Range;
//First Row Entry :
var rangeEntry1 = entries.Create<MappingTableEntryRange>(); //Create a new row
rangeEntry1.From = 0; // Add a From entry
rangeEntry1.To = 50; // Add a To Entry
rangeEntry1.Value = Color.Red; // Set the property value
rangeEntry1.Flashing = true; // Select the Flashing state
rangeEntry1.FlashingRate = FlashingRate.Medium; // Select the Flashing rate
rangeEntry1.AlternateValue = Color.Orange; // Select the Flashing color
//Second Row Entry :
var rangeEntry2 = entries.Create<MappingTableEntryRange>(); //Create a new row
rangeEntry2.From = 51; // Add a From entry
rangeEntry2.To = 80; // Add a To Entry
rangeEntry2.Value = Color.Yellow; // Set the property value
rangeEntry2.Flashing = true; // Select the Flashing state
rangeEntry2.FlashingRate = FlashingRate.Medium; // Select the Flashing rate
rangeEntry2.AlternateValue = Color.YellowGreen; // Select the Flashing color
//Third Row Entry :
var rangeEntry3 = entries.Create<MappingTableEntryRange>(); //Create a new row
rangeEntry3.From = 81; // Add a From entry
rangeEntry3.To = 100; // Add a To Entry
rangeEntry3.Value = Color.Green; // Set the property value
rangeEntry3.Flashing = false; // Select the Flashing state
//Fourth Row Entry :
var rangeEntry4 = entries.Create<MappingTableEntryRange>(); //Create a new row
rangeEntry4.From = 101; // Add a From entry
rangeEntry4.To = 101; // Add a To Entry
rangeEntry4.Value = Color.Green; // Set the property value
rangeEntry4.Flashing = true; // Select the Flashing state
rangeEntry4.FlashingRate = FlasingRate.Medium;
rangeEntry4.AlternateValue = Color.LightGreen;
//Second Row Entry - Bit number 4 :
var multibitEntries2 = entries.Create(BitDynamizationType.MultiBit);
var multibitEntry2 = multibitEntries2[0];
ulong multibitCondition2 = 16; // Condition = 2^(Bit No.) = 2^4 => 16
ulong multibitRelevant2 = 20; // Relevant = Decimal equivalent of no. of bits being
configured = 0001 0100 => 20
multibitEntry2.Update(multibitCondition2, multibitRelevant2);
multibitEntry2.Value = Color.Green;
multibitEntry2.Flashing = false; // Single Bit
var hmiSoftware = GetScadaHmiSoftware();
var Screen1 = hmiSoftware.Screens[0];
var screenItem0 = Screen1.ScreenItems.FirstOrDefault(x => x.Name == "Process control_1");
string propertyName = "Visible";
var tagDynamization = screenItem0.Dynamizations.Where(x => x.PropertyName ==
propertyName).FirstOrDefault() as TagDynamization;

```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

```

var mappingTable = tagDynamization.ValueConverter.MappingTable;
var entries = mappingTable.Entries;
var singleBitEntries = entries.Create(BitDynamizationType.SingleBit);
mappingTable.ConditionType = ConditionType.Singlebit; // Select Condition type
//State 0 for Bit 15 :
var singleBitEntry1 = singleBitEntries[0]; // Select the 1st Entry - Off State of the
bitulong singlebitCondition1 = 0; // Condition value = [Bit State x 2^ Bit number] => (0 x
2^15 = 0)ulong singlebitRelevant1 = 32768; // Relevant = Decimal equivalent of the Single
bit being configured = 1000 0000 0000 0000 => 32768
singleBitEntry1.Update(singlebitCondition1, singlebitRelevant1);
singleBitEntry1.Value = true; //State 1 for Bit 15 :
var singleBitEntry2 = singleBitEntries[1]; // Select the 2nd Entry - ON State of the bit
ulong singlebitCondition2 = 32768; // Condition value = [Bit State x 2^ Bit number] => (1 x
2^15 = 32768)
ulong singlebitRelevant2 = 32768; // Relevant = Decimal equivalent of the Single bit being
configured = 1000 0000 0000 0000 => 32768
singleBitEntry2.Update(singlebitCondition2, singlebitRelevant2);
singleBitEntry2.Value = false;

```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Beschreibung HMI Software (Seite 676)

5.13.2.4 Auf den Gerätenamen eines HMI-Panels zugreifen (RT Unified)

Einleitung

Sie können mit TIA Portal Openness die Gerätenamen von Unified und Nicht-Unified HMI-Geräten lesen und schreiben.

Eigenschaften

Auf die folgenden Attribute kann im HMI-Gerät zur Bearbeitung zugegriffen werden:

Attribut	Datentyp	Beschreibung	Zugriff
Name	String	Name des HMI-Geräts	R/W

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84).
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

```
Tiaproject = tiaPortal.Projects[0];
var deviceUnderTest = Tiaproject.Devices[0];
//Set
deviceUnderTest.Name = "Test1";
//Get
var deviceName = deviceUnderTest.Name;
foreach (var deviceItem in deviceUnderTest.DeviceItems)
{
    //Set
    deviceItem.Name = "TestDeviceItem1";
    //Get
    var deviceName = deviceItem.Name;
}
```

5.13.2.5 Zugriff auf Hierarchien im Common Plant Model (RT Unified)

Mit der Anlagensicht arbeiten (RT Unified)

Einleitung

Sie können die folgenden Aktionen für die Anlagensicht durchführen, während Sie TIA Portal Openness nutzen:

- Anlagensicht erstellen
- Anlagensicht löschen
- Anlagensicht umbenennen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um eine Anlagensicht zu erstellen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Create("ABCManufacturingPlant");
```

Hinweis

Sie können nur eine Anlagensicht innerhalb eines Projekts erstellen.

Ändern Sie folgenden Programmcode, um eine Anlagensicht zu löschen oder zu entfernen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Find("ABCManufacturingPlant");
plantView.Delete();
```

Ändern Sie folgenden Programmcode, um eine Anlagensicht durch Aktualisierung der Eigenschaft 'Name' umzubenennen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Find("ABCManufacturingPlant");
plantView.Name = "New_PlantView_Name";
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Mit Anlagensichtknoten arbeiten (RT Unified)**Einleitung**

Sie können die folgenden Aktionen für Anlagensichtknoten durchführen, während Sie TIA Portal Openness nutzen:

- Anlagensichtknoten erstellen
- Anlagensichtknoten löschen
- Anlagensichtknoten umbenennen

Eigenschaften

Die folgenden Eigenschaften werden in einem Anlagensichtknoten mit TIA Portal Openness unterstützt:

Name der Eigenschaft	Datentyp	Zugriffsmöglichkeit
HierarchyPath	String	R
PlantObject	Siemens.Engineering.HmiUnified.Cpm.PlantObject	R
Name	String	R/W
PlantView	String	R

Name der Eigenschaft	Datentyp	Zugriffsmöglichkeit
ViewPath	String	R
PlantViewNodes	PlantViewNodeComposition	R

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um einen Anlagensichtknoten zu erstellen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Find("ABCManufacturingPlant");
PlantViewNodeComposition viewNodes = plantView.PlantViewNodes;
PlantViewNode mixingNode = viewNodes.Create("Mixing");
```

Ändern Sie folgenden Programmcode, um einen Anlagensichtknoten unter einem anderen Anlagensichtknoten zu erstellen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Find("ABCManufacturingPlant");
PlantViewNodeComposition viewNodes = plantView.PlantViewNodes;
PlantViewNode mixingNode = viewNodes.Create("Mixing");
PlantViewNode motorNode = mixingNode.Create("Motor");
```

Hinweis

Sie können beliebig viele Anlagensichtknoten innerhalb einer Anlagensicht oder unter einem anderen Anlagensichtknoten erstellen.

Ändern Sie folgenden Programmcode, um einen Anlagensichtknoten zu entfernen oder zu löschen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Find("ABCManufacturingPlant");
PlantViewNode mixingNode = plantView.PlantViewNodes.Find("Mixing");
mixingNode.Delete();
```


5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Ändern Sie folgenden Programmcode, um einen Anlagensichtknoten durch Aktualisierung der Eigenschaft 'Name' umzubenennen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Find("ABCManufacturingPlant");
PlantViewNodeComposition viewNodes = plantView.PlantViewNodes;
PlantViewNode mixingNode = viewNodes.Create("Mixing");
mixingNode.Name = "New_PlantViewNode_Name";
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

Anlagensicht enumerieren (RT Unified)

Einleitung

Sie können mit TIA Portal Openness die in einem bestimmten Projekt im TIA Portal vorhandene Anlagensicht zu enumerieren.

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um alle Anlagensichten des Projekts zu enumerieren:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
foreach (var item in plantViews)
{
    // work with plant views (currently only 1 plant view can be created inside a project)
}
```

Ändern Sie folgenden Programmcode, um eine Anlagensicht in einer Liste der Anlagensichten anhand des Namens zu finden:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Find("ABCManufacturingPlant");
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Ändern Sie folgenden Programmcode, um eine Anlagensicht in einer Liste der Anlagensichten anhand des Index zu finden:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews[0];
```

Ändern Sie folgenden Programmcode, um einen Anlagensichtknoten in einer Liste der Anlagensichten anhand des Hierarchiepfads des gewünschten Knotens zu finden:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantViewNode plantViewNode =  
plantViews.FindPlantViewNode("ABCManufacturingPlan\Mixing\Motor");
```

Ändern Sie folgenden Programmcode, um einen Anlagensichtknoten in einer Liste der Anlagensichtknoten anhand des Namens der Anlagensicht zu finden:

```
PlantViewNodeComposition plantViewNodes =  
m_tiaPortalApp.Projects[0].PlantViews[0].PlantViewNodes;  
PlantViewNode plantViewNode = plantViewNodes.Find("Mixing");
```

Ändern Sie folgenden Programmcode, um alle Listen von Knoten in Anlagensichten, die innerhalb der Anlagensicht verfügbar sind, zu enumerieren:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantViewNodeComposition nodes = plantView.PlantViewNodes;  
foreach (var item in nodes)  
{  
    // work with plant view nodes inside plant view  
}
```

Ändern Sie folgenden Programmcode, um alle Knoten in Anlagensichten, die innerhalb sämtlicher Hierarchiestufen verfügbar sind, zu enumerieren:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantViewNodeComposition nodes = plantView.PlantViewNodes;  
PlantViewNodeComposition subNodes = nodes[2].PlantViewNodes;  
foreach (var item in subNodes)  
{  
    // work with plant view nodes inside 2nd node within the plant view  
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Mit der Anlagensicht und Geräten arbeiten (RT Unified)

Einleitung

Sie können folgende Aktionen für den Zugriff auf Hierarchien in Anlagensichten und Anlagenobjekte durchführen, während Sie TIA Portal Openness verwenden:

- Gerät einer Anlagensicht zuordnen
- Gerät einer Anlagensicht ändern
- Gerät aus einer Anlagensicht entfernen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um durch Festlegen der Eigenschaft 'AssignedHmiDevice' ein Gerät der Anlagensicht zuzuordnen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews.Create("ABCManufacturingPlant");  
plantView.AssignedHmiDevice = "HMI_RT_1";
```

Ändern Sie folgenden Programmcode, um durch Festlegen eines Gerätenamens in der Eigenschaft 'AssignedHmiDevice' ein Gerät in der Anlagensicht zu ändern:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews.Create("ABCManufacturingPlant");  
plantView.AssignedHmiDevice = "HMI_RT_2";
```

Ändern Sie folgenden Programmcode, um durch Festlegen eines leeren Gerätenamens in der Eigenschaft 'AssignedHmiDevice' ein Gerät in der Anlagensicht zu entfernen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews.Create("ABCManufacturingPlant");  
plantView.AssignedHmiDevice = string.Empty;
```

Hinweis

Um das Gerät zu entfernen, verwenden Sie nur `string.empty`. NULL oder Leerzeichen (" ") wird als Geräteiname erkannt und erzeugt eine Fehlermeldung, dass das Gerät nicht gefunden wurde.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.2.6 Zugriff auf Instanzen im Common Plant Model (RT Unified)**Mit CPM-Objektinstanzen arbeiten (RT Unified)****Einleitung**

Sie können folgende Aktionen für den Zugriff auf eine CPM-Instanz anhand des CPM-Typs durchführen, während Sie TIA Portal Openness verwenden:

- CPM-Objektinstanzen erstellen
- CPM-Objektinstanzen löschen
- CPM-Objektinstanzen umbenennen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Eine Anlage ist erstellt

Programmcode

Ändern Sie folgenden Programmcode, um eine CPM-Objektinstanz zu erstellen:

```
// Instance inside plant view
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews.Create("ABCManufacturingPlant");
plantView.AssignedHmiDevice = "HMI_RT_1";
PlantViewNodeComposition nodes = plantView.PlantViewNodes;
PlantViewNode node = nodes.Create("ObjectLevel1", "Plant_Object_Type_1");
//Instance inside plant view node
PlantViewNode mixerObject =
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Create("IngredientMixer",
"Plant_Object_Type_1");
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Ändern Sie folgenden Programmcode, um eine CPM-Objektinstanz zu entfernen oder zu löschen:

```
PlantViewNode mixingNode = plantView.PlantViewNodes.Find("Mixing");  
PlantViewNode mixerObject =  
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Find("IngredientMixer");  
mixerObject.Delete();
```

Ändern Sie folgenden Programmcode, um durch Aktualisierung der Eigenschaft 'Name' eine CPM-Objektinstanz umzubenennen:

```
PlantViewNode mixingNode = plantView.PlantViewNodes.Find("Mixing");  
PlantViewNode mixerObject =  
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Find("IngredientMixer");  
mixerObject.Name = "New_PlantViewNode_Name";
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Mit Anlagensichtknoten von CPM-Objektinstanzen arbeiten (RT Unified)

Einleitung

Sie können die folgenden Aktionen für den Zugriff auf Anlagensichtknoten von CPM-Objektinstanzen durchführen, während Sie TIA Portal Openness nutzen:

- Anlagensichtknoten erstellen
- Anlagensichtknoten löschen
- Anlagensichtknoten umbenennen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um einen Anlagensichtknoten innerhalb einer CPM-Objektinstanz zu erstellen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews.Find("ABCManufacturingPlant");  
PlantViewNode mixerObject = plantView.PlantViewNodes.Create("ObjectLevel1",  
"Plant_Object_Type_1");  
PlantViewNode mixingNode = mixerObject.PlantViewNodes.Create("Mixing");
```

Hinweis

Sie können beliebig viele Anlagensichtknoten innerhalb eines CPM-Objekts erstellen.

Ändern Sie folgenden Programmcode, um eine CPM-Objektinstanz innerhalb eines Anlagensichtknotens zu erstellen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews.Find("ABCManufacturingPlant");  
PlantViewNode mixingNode = plantView.PlantViewNodes.Create("Mixing");  
PlantViewNode mixerObject = mixingNode.PlantViewNodes.Create("ObjectLevel1",  
"Plant_Object_Type_1");
```

Hinweis

Sie können beliebig viele CPM-Objektinstanzen innerhalb eines Anlagensichtknotens erstellen.

Ändern Sie folgenden Programmcode, um einen Anlagensichtknoten aus einem Anlagensichtknoten zu entfernen oder zu löschen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews.Find("ABCManufacturingPlant");  
PlantViewNode mixerObject = plantView.PlantViewNodes.Find("ObjectLevel1");  
PlantViewNode mixingNode = mixerObject.PlantViewNodes.Find("Mixing");  
mixingNode.Delete();
```

Ändern Sie folgenden Programmcode, um durch Aktualisierung der Eigenschaft 'Name' einen Anlagensichtknoten einer CPM-Objektinstanz umzubenennen:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews.Find("ABCManufacturingPlant");  
PlantViewNode mixerObject = plantView.PlantViewNodes.Find("ObjectLevel1",  
"Plant_Object_Type_1");  
PlantViewNode mixingNode = mixerObject.PlantViewNodes.Find("Mixing");  
mixingNode.Name = "New_PlantViewNode_Name";
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Schnittstellenvariablen von CPM-Objektinstanzen enumerieren (RT Unified)

Einleitung

Sie können mit TIA Portal Openness ein Member einer CPM-Objektinstanz enumerieren.

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um alle Members der CPM-Objektinstanz zu enumerieren:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews[0];  
PlantViewNodeComposition plantViewNodes = plantView.PlantViewNodes;  
PlantViewNode plantViewNode = plantViewNodes.Find("MixerObject_1");  
PlantObject plantObject = plantViewNode.PlantObject;  
PlantObjectInterfaceComposition interfaces = plantObject?. PlantObjectInterfaces;
```

Hinweis

Im Beispiel oben bezieht sich der Operator ?. auf eine Nullprüfung.

Ändern Sie folgenden Programmcode, um eine Schnittstellenvariable in der Liste der Schnittstellen anhand des Namens zu finden:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;  
PlantView plantView = plantViews[0];  
PlantViewNodeComposition plantNodes = plantView.PlantViewNodes;  
PlantViewNode plantViewNode = plantNodes.Find("MixerObject_1");  
PlantObject plantObject = plantViewNode.PlantObject;  
PlantObjectInterface interface = plantObject.PlantObjectInterfaces.Find("Interface_1");
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Ändern Sie folgenden Programmcode, um eine Schnittstellenvariable in der Liste der Schnittstellen anhand des Index zu finden:

```
PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews[0];
PlantViewNodeComposition plantViewNodes = plantView.PlantViewNodes;
PlantViewNode plantViewNode = plantViewNodes.Find("MixerObject_1");
PlantObject plantObject = plantViewNode.PlantObject;
PlantObjectInterfaceComposition interfaces = plantObject.PlantObjectInterfaces;
PlantObjectInterface interface = interfaces[0];
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.2.7 Auf Eigenschaften für Schnittstellen/Archivvariablen von Anlagenobjektinstanzen zugreifen (RT Unified)

Zugriff auf und Aktualisierung von Eigenschaften von Schnittstellenvariablen von CPM-Anlagenobjektinstanzen (RT Unified)

Einleitung

Sie können mit TIA Portal Openness auf die Eigenschaften von Schnittstellenvariablen von CPM-Anlagenobjektinstanzen zugreifen und diese aktualisieren.

Auf die folgenden Eigenschaften von Schnittstellenvariablen von CPM-Anlagenobjektinstanzen kann zugegriffen werden:

Name der Eigenschaft	Datentyp	Beschreibung	Zugänglichkeit
Name	System.String	Name der Schnittstellenvariablen	R/W
PlcTag	System.String	PLC-Variable zu Schnittstellenvariablen	R/W
Connection	System.String	Anschaltung von Schnittstellenvariablen	R/W
PlcName	System.String	PLC-Name zu Schnittstellenvariablen	R
AccessMode	PlantObjectTagAccessMode	Zugriffsmodus für Schnittstellenvariablen	R
DataType	System.String	Objekttyp der Schnittstellenvariablen	R
MaxLength	System.Int	Länge von Schnittstellenvariablen	R
HmiDataType	System.Int	HMI-Datentyp von Schnittstellenvariablen	R

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugänglichkeit
AcquisitionMode	PlantObjectTagAcquisitionMode	Erfassungsart von Schnittstellenvariablen	R/W
AcquisitionCycle	System.Int	Erfassungszyklus für Schnittstellenvariablen	R/W
Persistent	System.Bool	Persistenz interner Variablen der CPM-Objektinstanz	R
Comment	System.String	Kommentar	R/W
Members	PlantObjectInterfaceMemberComposition	Mitglieder der Schnittstellenvariable	R

Hinweis

Die Eigenschaft PLCTag & Connection kann erst konfiguriert werden, nachdem das HMI-Gerät der Anlagensicht zugeordnet wurde.

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um auf alle Eigenschaften von Schnittstellenvariablen zuzugreifen:

```
// Instance inside plant view node
PlantViewNode mixingNode = plantView.PlantViewNodes.Create("Mixing");
PlantViewNode mixerObject =
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Create("IngredientMixer", "Mixer");
PlantObjectInterfaceComposition interfaceTags =
mixerObject.PlantObject.PlantObjectInterfaces;
PlantObjectInterface interfaceTag = interfaceTags.FirstOrDefault();
string name = interfaceTag.Name;
MultilingualText comment = interfaceTag.Comment
```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Ändern Sie folgenden Programmcode, um einige Eigenschaften einer Schnittstellenvariable zu aktualisieren, z. B. Kommentar, PLC-Variable, Anschaltung, Erfassungsmodus, Erfassungszyklus:

```
// Instance inside plant view node
PlantViewNode mixingNode = plantView.PlantViewNodes.Create("Mixing");
PlantViewNode mixerObject =
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Create("IngredientMixer", "Mixer");
PlantObjectInterfaceComposition interfaceTags =
mixerObject.PlantObject.PlantObjectInterfaces;
PlantObjectInterface interfaceTag = interfaceTags.FirstOrDefault();
interfaceTag.Name = "New_InterfaceTag_Name";
interfaceTag.Comment.Items[0].Text = "New_InterfaceTag_Comment";
interfaceTag.AcquisitionMode = PlantObjectTagAcquisitionMode.CyclicContinuous;
interfaceTag.AcquisitionCycle = "T12s";
interfaceTag.Connection = "Connection_1";
interfaceTag.PlcTag = "PLC_Tag_1";
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Zugriff auf und Aktualisierung von Eigenschaften von Member-Variablen von Schnittstellenvariablen (RT Unified)

Einleitung

Sie können mit TIA Portal Openness auf die Eigenschaften von Member-Variablen von Schnittstellenvariablen zugreifen und diese aktualisieren.

Auf die folgenden Eigenschaften von Member-Variablen von Schnittstellenvariablen kann zugegriffen werden:

Name der Eigenschaft	Datentyp	Beschreibung	Zugänglichkeit
Name	System.String	Name der Member-Variablen	R
DataType	System.String	Objektyp der Member-Variablen	R
MaxLength	System.Int	Länge der Member-Variablen	R
HmiDataType	System.String	HMI-Datentyp der Member-Variablen	R
InitialMinValue	PlantObjectTagLowerRange	Unterer Bereichswert	R/W
InitialMaxValue	PlantObjectTagUpperRange	Oberer Bereichswert	R/W
InitialValue	System.Object	Anfangswert	R/W
SubstituteValue	PlantObjectTagSubstituteValue	Ersatzwert	R
Comment	MultilingualText	Kommentar	R/W

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Name der Eigenschaft	Datentyp	Beschreibung	Zugänglichkeit
Members	PlantObjectInterfaceMemberComposition	Mitglieder der Member-Variable	R
LoggingTags	PlantObjectLoggingTagComposition	Archivvariablen der Member-Variable	R

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode:

```

PlantViewComposition plantViews = m_tiaPortalApp.Projects[0].PlantViews;
PlantView plantView = plantViews[0];
PlantViewNodeComposition plantViewNodeComposition = plantView.PlantViewNodes;
PlantViewNode plantViewNode = plantViewNodeComposition[0];
PlantObjectInterfaceComposition tagComposition =
plantViewNode.PlantObject.PlantObjectInterfaces;
PlantObjectInterface interfaceTag = tagComposition[0];
PlantObjectInterfaceMemberComposition memberTags = interfaceTag.Members;
PlantObjectInterfaceMember memberTag = memberTags.FirstOrDefault();
//GetAttributes
List<string> listOfPropertyNames = new List<string>()
{
    "Name"
};
var listOfPropertyValues = memberTag.GetAttributes(listOfPropertyNames);
//SetAttributes
IEnumerable<KeyValuePair<string, object>> propertyNameValuePairList = new
List<KeyValuePair<string, object>>()
{
    new KeyValuePair<string, object>("Name", "MemberTag_1");
};
memberTag.SetAttributes(propertyNameValuePairList);

```

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Ändern Sie folgenden Programmcode, um auf alle Eigenschaften der Member-Variablen zuzugreifen:

```
// Instance inside plant view node
PlantViewNode mixingNode = plantView.PlantViewNodes.Create("Mixing");
PlantViewNode mixerObject =
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Create("IngredientMixer", "Mixer");
PlantObjectInterfaceComposition interfaceTags =
mixerObject.PlantObject.PlantObjectInterfaces;
PlantObjectInterfaceMemberComposition memberTags = interfaceTag.Members;
PlantObjectInterfaceMember memberTag = memberTags.FirstOrDefault();
string name = memberTag.Name;
string comment = memberTag.Comment.Items[0].Text;
```

Ändern Sie folgenden Programmcode, um Eigenschaften einer Member-Variablen einer CPM-Objektinstanz zu aktualisieren:

```
// Instance inside plant view node
PlantViewNode mixingNode = plantView.PlantViewNodes.Create("Mixing");
PlantViewNode mixerObject =
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Create("IngredientMixer", "Mixer");
PlantObjectInterfaceComposition interfaceTags =
mixerObject.PlantObject.PlantObjectInterfaces;
PlantObjectInterfaceMember memberTag = memberTags.FirstOrDefault();
memberTag.Comment.Items[0].Text = "New_MemberTag_Comment";
memberTag.InitialValue = 2;
memberTag.InitialMinValue.Value = 2;
memberTag.InitialMinValue.ValueType = PlantObjectTagLimitValueType.Constant;
memberTag.InitialMaxValue.Value = 2;
memberTag.InitialMaxValue.ValueType = PlantObjectTagLimitValueType.Constant;
```

Ändern Sie folgenden Programmcode, um auf alle Archivvariablen einer Member-Variablen einer CPM-Objektinstanz zuzugreifen:

```
// Instance inside plant view node
PlantViewNode mixingNode = plantView.PlantViewNodes.Create("Mixing");
PlantViewNode mixerObject =
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Create("IngredientMixer", "Mixer");
PlantObjectInterfaceComposition interfaceTags =
mixerObject.PlantObject.PlantObjectInterfaces;
PlantObjectInterfaceMember memberTag = memberTags.FirstOrDefault();
PlantObjectInterfaceMemberComposition memberTags = interfaceTag.Members;
PlantObjectInterfaceMember memberTag = memberTags.FirstOrDefault();
PlantObjectLoggingTagComposition loggingTags = memberTag.LoggingTags;
PlantObjectLoggingTag loggingTag = loggingTags.FirstOrDefault();
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Zugriff auf und Aktualisierung von Eigenschaften von Logging-Variablen von Member-Variablen (RT Unified)

Einleitung

Sie können mit TIA Portal Openness auf die Archivvariablen von Member-Variablen in CPM-Anlageobjektinstanzen zugreifen.

Auf die folgenden Eigenschaften von Archivvariablen von Member-Variablen in CPM-Anlageobjektinstanzen kann zugegriffen werden:

Name der Eigenschaft	Datentyp	Beschreibung	Zugänglichkeit
AggregationDelay	System.TimeSpan	Wert für Aggregationsverzögerung	R
AggregationMode	PlantObjectLoggingTagAggregationMode	Wert für Aggregationsmodus	R
Cycle	System.String	Archivierungszyklus der Archivvariable	R
CycleFactor	System.UInt32	Wert für Archivierungszyklusfaktor	R
DataLog	System.String		R
Source	System.string	Quelle der Archivvariable	R
TriggerMode	PlantObjectLoggingTagTriggerMode	Auslösemodus der Archivvariable	R
TriggerTag	System.String	Wert der Auslösevariable	R
TriggerTagBitNumber	System.UInt32	Wert von TriggerTagBitNumber	R
Name	System.string	Name der Archivvariable	R
LogConfiguration	System.String	Datenarchiv der Archivvariable	R
LoggingMode	PlantObjectLoggingTagLoggingMode	Protokollierungsmodus der Archivvariable	R
SmoothingMode	PlantObjectLoggingTagSmoothingMode	Glättungsmodus der Archivvariable	R
SmoothingMinTime	System.TimeSpan	Mindestzeit der Archivvariable	R
SmoothingMaxTime	System.TimeSpan	Höchstzeit der Archivvariable	R
SmoothingDeltaValue	System.Double	Delta der Archivvariable	R
LimitScope	PlantObjectLoggingTagLimitScope	Umfangsbegrenzung der Archivvariable	R
HighLimit	System.Object	Oberer Grenzwert der Archivvariable	R
LowLimit	System.Object	Unterer Grenzwert der Archivvariable	R

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um auf alle Eigenschaften von Archivvariablen zuzugreifen:

```
// Instance inside plant view node
PlantViewNode mixingNode = plantView.PlantViewNodes.Create("Mixing");
PlantViewNode mixerObject =
plantView.PlantViewNodes.Find("Mixing").PlantViewNodes.Create("IngredientMixer", "Mixer");
PlantObjectInterfaceMemberComposition interfaceTags =
mixerObject.PlantObject.PlantObjectInterfaceMembers;
PlantObjectInterfaceMember interfaceTag = interfaceTags.FirstOrDefault();
PlantObjectInterfaceMemberComposition memberTags = interfaceTag.Members;
PlantObjectInterfaceMember memberTag = memberTags.FirstOrDefault();
PlantObjectLoggingTagComposition loggingTags = memberTag.LoggingTags;
PlantObjectLoggingTag loggingTag = loggingTags.FirstOrDefault();
string name = loggingTag.Name;
object lowLimit = loggingTag.LowLimit;
```

Hinweis

Eigenschaften der Klasse PlantObjectLoggingTag zur Archivvariablen einer CPM-Objektinstanz können nicht aktualisiert werden.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.13.2.8 Lizenz für den Zugriff auf ein Unified-Gerät prüfen (RT Unified)

Einleitung

Über die TIA Openness-Schnittstelle können Sie jede Openness Client-Anwendung auf das Vorhandensein einer WinCC Unified-Lizenz überprüfen, so dass eine Client-Anwendung die Lizenzierung nicht durch Verwendung von Openness-Schnittstellen umgehen kann.

Sie können ein HMI Unified-Gerät und seine unterlagerten Objekte (z.B. Variablen, Meldungen, Bilder usw.) nur dann anzeigen, wenn eine HMI Unified-Lizenz vorhanden ist.

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Jede andere Aktion auf Geräteebe (z.B. Erstellen, Suchen, Löschen) führt zur Rückgabe einer Ausnahme `LicenseNotFoundException` von der Openness API.

Hinweis

Bei HMI Unified Software wird bei einer ungültigen Lizenz die Ausnahme `LicenseNotFoundException` zurückgegeben.

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Es besteht Zugriff auf das HMI-Softwareobjekt
Siehe HMI Unified-Software-Objekt (Seite 676)

Programmcode

Ändern Sie den folgenden Programmcode, um auf das Gerät oder Teilsysteme des Geräts zuzugreifen, wenn keine Lizenz vorhanden ist; es wird eine Ausnahme `LicenseNotFound` zurückgegeben:

```
private static void ListDevices(Project project)
{
    foreach (var device in project.Devices)
    {
        ListDeviceItem(device);
    }
}

private static void ListDeviceItem(Device device)
{
    try
    {
        Console.WriteLine("HMI device - " + device.Name);
        foreach (var item in device.DeviceItems)
        {
            Console.WriteLine("HMI device type - " + item.TypeIdentifier);
            var softContainer = item.GetService<SoftwareContainer>();
            if (softContainer != null)
            {
                var softTarget = softContainer.Software;
            }
            if (softTarget != null)
            {
                Console.WriteLine("HMI device software - " + softTarget.Name);
            }
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }
}
```

Hinweis

Während des Zugriffs auf obigen Programmcode ist das `SoftTarget` null und es ist kein Zugriff auf Teilsysteme wie Alarmer oder Variablen möglich.

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

Ändern Sie folgenden Programmcode, um auf das Gerät oder Teilsysteme des Geräts zuzugreifen, wenn eine Lizenz vorhanden ist; es wird ein Nullwert zurückgegeben:

```
private static void ListDevices(Project project)
{
    foreach (var device in project.Devices)
    {
        ListDeviceItem(device);
    }
}

private static void ListDeviceItem(Device device)
{
    Console.WriteLine("HMI device - " + device.Name);
    foreach (var item in device.DeviceItems)
    {
        Console.WriteLine("HMI device type - " + item.TypeIdentifier);
        var softContainer = item.GetService<SoftwareContainer>();
        if (softContainer != null)
        {
            var softTarget = softContainer.Software;
        }
        if (softTarget != null)
        {
            Console.WriteLine("HMI device software - " + softTarget.Name);
        }
    }
}
```

Hinweis

Während des Zugriffs auf obigen Programmcode enthält das SoftTarget Gerätebehälter und der Zugriff auf Teilsysteme wie Alarmer und Variablen ist möglich.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Beschreibung HMISoftware (Seite 676)

5.13.2.9 Querverweise für WinCC Unified abrufen (RT Unified)**Einleitung**

Sie können mit TIA Portal Openness Querverweisinformationen zu anwendbaren WinCC Unified-Objekten bereitstellen. Die Openness API gibt das CrossReferenceResult mit den Textwerten der Benutzeroberfläche 'Querverweise' und das tatsächliche IEngineeringObject der anwendbaren WinCC Unified-Objekte zurück.

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

'Querverweise' versucht, das WinCC Unified-Objekt selbst bereitzustellen, für das Sie, unterstützt durch die jeweiligen TIA Portal Openness-Objekte, zusätzliche Aktionen ausführen können.

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Zugriff auf HMI-Softwareobjekt
Siehe HMI-Unified-Software (Seite 676)

Eigenschaften

Eine Liste mit von WinCC Unified-Objekten unterstützten Eigenschaften finden Sie unter Eigenschaften, Querverweise für STEP 7 aufrufen (Seite 464).

Programmcode

```

Public void GetCrossReferences()
{
    CrossReferenceService crossReferenceService =
    HmiScreen.GetService<CrossReferenceService>();
    CrossReferenceResult rootResultObject =
    crossReferenceService.GetCrossReferences(CrossReferenceFilter.AllObjects);
    SourceObjectComposition rootResultSourceObjects = rootResultObject.Sources; //The API
    returns a single source object but the type is a composition.
    foreach (SourceObject sourceObject in rootResultSourceObjects)
    {
        //Get the source object references
        ReferenceObjectComposition referenceObjects = source.References;
        PrintReferences(referenceObjects);
    }
}

Private void PrintReferences(ReferenceObjectComposition referenceObjects)
{
    foreach (ReferenceObject referenceObject in referenceObjects)
    {
        // Needs similar casting for each type as seen in source object example snippet
        if (source.UnderlyingObject != null && source.UnderlyingObject is HmiScreenItemBase)
        {
            var hmiScreenItem= source.UnderlyingObject as HmiScreenItem;
            //Perform actions on hmiScreenItem.
        }
        //No other attributes are available from the IEngineeringObject. All the remaining values
        must be given from crossreference separately via ReferenceObject EOM.
        Console.WriteLine(referenceObject.Name);
        Console.WriteLine(referenceObject.Address);
        Console.WriteLine(referenceObject.TypeName);
        Console.WriteLine(referenceObject.Device);
        Console.WriteLine(referenceObject.Path);
        LocationComposition locations = referenceObject.Locations;
        //Looping through ReferenceLocations
        {
            Console.WriteLine(location.Name);
            Console.WriteLine(location.ReferenceLocation);
            Console.WriteLine(location.ReferenceType);
            Console.WriteLine(location.Access);
            if(location.ReferencedAs is Tag)
            {
                var referencedAsObject = location.ReferencedAs as Tag;
            }
            // Needs similar casting for each type as seen in source object example snippet
            Console.WriteLine(location.ReferncedAsName);
            Console.WriteLine(location.Address);
            Console.WriteLine(location.TypeName);
        }
    }
}

```

Siehe auch

- Querverweise für STEP 7 abrufen (Seite 464)
- Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen (Seite 131)
- Beschreibung HMISoftware (Seite 676)

5.13.2.10 Mit Dynamisierung und Ereignissen für Bilder/Bildobjekte über Skripte arbeiten (RT Unified)

Einleitung

Sie können folgende Aktionen mit Skripten zur Konfiguration der Eigenschaften-Dynamisierung und von Ereignissen für Bilder/Bildobjekte durchführen, während Sie TIA Portal Openness verwenden:

- Skript-Dynamisierungen erstellen
- Skript-Dynamisierungen enumerieren
- Skript-Dynamisierungen löschen
- Auf Eigenschaften von Skript-Dynamisierungen zugreifen
- Auf Eigenschaften eines Triggers zugreifen
- Syntax prüfen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Es besteht Zugriff auf das HMI-Unified-Softwareobjekt
Siehe HMI-Unified-Software-Objekt (Seite 676)

Programmcode

Bei der Arbeit mit der Skript-Dynamisierung für Bilder/Bildobjekte können Sie den folgenden Beispiel-Programmcode ändern und verwenden.

Skript-Dynamisierungen erstellen

Ändern Sie folgenden Programmcode, um die Skript-Dynamisierungen für eine Eigenschaft zu erstellen:

```
public DynamizationBase Create<DynamizationBase>(string propertyName);
HmiScreen screen = m_hmiSoftware.Screen[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
ScriptDynamization scriptDynamic = dyns.Create<ScriptDynamization>("BackColor");
```

Skript-Dynamisierungen enumerieren

Ändern Sie folgenden Programmcode, um die Skript-Dynamisierung einer Eigenschaft zu enumerieren:

```
public DynamizationBase Find(string propertyName);
HmiScreen screen = m_hmiSoftware.Screens[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
if (dyns is ScriptDynamization)
{
    ScriptDynamization scriptDynamic = (ScriptDynamization)dyns.Find("BackColor");
}
```

Skript-Dynamisierungen löschen

Ändern Sie folgenden Programmcode, um die Skript-Dynamisierung für eine Eigenschaft zu löschen:

```
HmiScreen screen = m_hmiSoftware.Screens[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
if (dyns is ScriptDynamization)
    ScriptDynamization scriptDynamic = (ScriptDynamization)dyns.Find("BackColor");
scriptDynamic.Delete();
```

Auf Eigenschaften von Skript-Dynamisierungen zugreifen

Ändern Sie folgenden Programmcode, um Eigenschaften der Skript-Dynamisierung abzurufen und festzulegen:

```
HmiScreen screen = m_hmiSoftware.Screens[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
foreach (DynamizationBase dynamic in dyns)
{
    if (dynamic.DynamizationType == DynamizationType.Script)
    {
        ScriptDynamization scriptDynamization = (ScriptDynamization)dynamic;
        scriptDynamization.Async = true;
        scriptDynamization.GlobalDefinitionAreaScriptCode = @"Add(parameter1,parameter2)";
        scriptDynamization.ScriptCode = @"
var value;
let tag1 = Tags('Tag_1');
let tagValue1 = tag1.Read();
HMIRuntime.Trace('value of MyTag1: ' + tagValue1);
return value; ";
        Trigger triggerObj = scriptDynamization.Trigger;
    }
}
```

Auf Eigenschaften eines Triggers zugreifen

Ändern Sie folgenden Programmcode, um Eigenschaften eines Triggers abzurufen und festzulegen:

```
HmiScreen screen = m_hmiSoftware.Screens[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
foreach (DynamizationBase dynamic in dyns)
{
    if (dynamic.DynamizationType == DynamizationType.Script)
    {
        ScriptDynamization scriptDynamization = (ScriptDynamization)dynamic;
        // Property write
        scriptDynamization.Trigger.Type = TriggerType.CustomCycle;
        scriptDynamization.Trigger.CustomDuration = "T500ms";
        scriptDynamization.Trigger.Tags = new List<string>() { "Tag_1" };
        // Property read
        Console.WriteLine(scriptDynamization.Trigger.Type);
        Console.WriteLine(scriptDynamization.Trigger.CustomDuration);
        Console.WriteLine(scriptDynamization.Trigger.Tags);
    }
}
```

Syntaxprüfung

Ändern Sie folgenden Programmcode, um mit der Skriptdynamisierungsaktion (Methode) `SyntaxCheck` die Syntax von Skript-Code oder von Code eines globalen Skripts zu prüfen.

```
HmiScreen screen = m_hmiSoftware.Screens[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
foreach (DynamizationBase dynamic in dyns)
{
    if (dynamic.DynamizationType == DynamizationType.Script)
    {
        ScriptDynamization scriptDynamization = (ScriptDynamization)dynamic;
        scriptDynamization.ScriptCode = @"
var value;let tag1 = Tags('Tag_1');
let tagValue1 = tag1.Read();
HMIRuntime.Trace('value of MyTag1: ' + tagValue1);
return value; ";
        HmiValidationResult syntaxCkResult = scriptDynamization.SyntaxCheck();
        IEnumerable<string> error = syntaxCkResult.Errors;
        IEnumerable<string> warnings = syntaxCkResult.Warning;
    }
}
```

Hinweis

Create - Recoverable exception wird ausgelöst, wenn die betreffende Dynamisierung für die angegebene Eigenschaft nicht unterstützt wird. Wenn während des Festlegens von Eigenschaften der Schreibvorgang die Konsistenzprüfungen nicht besteht, wird eine Recoverable exception ausgelöst.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Beschreibung HMI Software (Seite 676)

5.13.2.11 Mit Dynamisierung für Bilder/Bildobjekte arbeiten (RT Unified)

Einleitung

Sie können folgende Aktionen mit der Dynamisierungsfunktion von Bildern/Bildobjekten durchführen, während Sie TIA Portal Openness verwenden:

- Dynamisierung für eine Eigenschaft erstellen
- Dynamisierung für eine Eigenschaft enumerieren
- Dynamisierung für eine Eigenschaft löschen
- Auf gemeinsame Eigenschaften der Dynamisierung zugreifen
- Auf Eigenschaften der Variablen-Dynamisierung zugreifen

5.13 Funktionen für den Zugriff auf die Daten eines HMI Unified Gerätes (RT Unified)

- Auf Eigenschaften der Blink-Dynamisierung zugreifen
- Auf Eigenschaften der Ressourcenlisten-Dynamisierung zugreifen

Hinweis

Um auf die Dynamisierung für eine Eigenschaft auf Part-Objekt-Ebene zuzugreifen, müssen Sie auf das entsprechende Part-Objekt der Controls zugreifen. Informationen zu Parts finden Sie unter Auf Eigenschaften einer Kurvenanzeige zugreifen (Seite 84)

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Es besteht Zugriff auf das HMI-Softwareobjekt
Siehe HMI-Unified-Software-Objekt (Seite 676)

Programmcode

Bei der Arbeit mit Dynamisierungen für Bilder/Bildobjekte können Sie den folgenden Beispiel-Programmcode ändern und verwenden.

Dynamisierung für eine Eigenschaft erstellen

Ändern Sie folgenden Programmcode, um die Dynamisierung für eine Eigenschaft zu erstellen:

```
HmiScreen screen = m_hmiSoftware.Screens[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
// Flashing dynamization
FlashingDynamization FlashingDyn = dyns.Create<FlashingDynamization>("BackColor");
// ResourceList dynamization
ResourceListDynamization ResListDyn = dyns.Create<ResourceListDynamization>("DisplayName");
// Tag dynamization
TagDynamization tagDyn = dyns.Create<TagDynamization>("Width");
```


Dynamisierung für eine Eigenschaft enumerieren

Ändern Sie folgenden Programmcode, um die Dynamisierung für eine Eigenschaft zu enumerieren:

```
HmiScreen screen = m_hmiSoftware.Screens[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
// Flashing dynamization
FlashingDynamization FlashDyn = (FlashingDynamization)dyns.Find("BackColor");
// ResourceList dynamization
ResourceListDynamization resourceDyn = (ResourceListDynamization)dyns.Find("DisplayName");
// Tag dynamization
TagDynamization tagDyn = (TagDynamization)dyns.Find("Width");
```

Dynamisierung für eine Eigenschaft löschen

Ändern Sie folgenden Programmcode, um die Dynamisierung einer Eigenschaft zu löschen:

```
public void Delete ()
HmiScreen screen = m_hmiSoftware.Screens[0];
DynamizationBaseComposition dyns = screen.Dynamizations;
DynamizationBase dynamization = dyns.Find("BackColor");
dynamization.Delete();
```

Auf gemeinsame Eigenschaften der Dynamisierung zugreifen

Ändern Sie folgenden Programmcode, um gemeinsame Eigenschaften der Dynamisierung abzurufen:

```
HmiSoftware m_hmiSoftware = ..;
DynamizationBaseComposition screenDynamizations = m_hmiSoftware.Screens[0].Dynamizations;
DynamizationBase screenDynamization = screenDynamizations.Find("BackColor");
string propertyName = screenDynamization.PropertyName;
DynamizationType dynamizationType = screenDynamization.DynamizationType;
```

Auf Eigenschaften der Variablen-Dynamisierung zugreifen

Ändern Sie folgenden Programmcode, um Eigenschaften der Variablen-Dynamisierung abzurufen und festzulegen:

```
HmiScreen screen = m_hmiSoftware.Screens[0];
TagDynamization tagDynamization = (TagDynamization)screen.Dynamizations.Find("Width");
foreach (DynamizationBase dynamization in screen.Dynamizations)
{
    if (dynamization.DynamizationType == DynamizationType.Tag)
    {
        TagDynamization TagDynamization = (TagDynamization)dynamization;
        TagDynamization.Tag = "HmiTagName";
        TagDynamization.UseIndirectAddressing = true;
        TagDynamization.ReadOnly = true;
    }
}
```

Auf Eigenschaften der Blink-Dynamisierung zugreifen

Ändern Sie folgenden Programmcode, um Eigenschaften der Blink-Dynamisierung abzurufen und festzulegen:

```
HmiScreen screen = m_hmiSoftware.Screens[0];
foreach (DynamizationBase dynamization in screen.Dynamizations)
{
    if (dynamization.DynamizationType == DynamizationType.Flashing)
    {
        FlashingDynamization flashingDynamization = (FlashingDynamization)dynamization;
        flashingDynamization.AlternateColor = Color.Beige;
        flashingDynamization.Color = Color.FromArgb(12, 22, 152, 200);
        flashingDynamization.FlashingCondition = FlashingCondition.Always;
        flashingDynamization.FlashingRate = FlashingRate.Medium;
    }
}
```

Auf Eigenschaften der Ressourcenlisten-Dynamisierung zugreifen

Ändern Sie folgenden Programmcode, um Eigenschaften der Ressourcenlisten-Dynamisierung abzurufen und festzulegen

```
HmiScreen screen = m_hmiSoftware.Screens[0];
foreach (DynamizationBase dynamization in screen.Dynamizations)
{
    if (dynamization.DynamizationType == DynamizationType.ResourceList)
    {
        ResourceListDynamization resourceListDyn = (ResourceListDynamization)dynamization;
        resourceListDyn.Tag = "Tag name";
        resourceListDyn.ResourceList = "Resource list name";
    }
}
```

Hinweis

Create - Recoverable exception wird ausgelöst, wenn die betreffende Dynamisierung für die angegebene Eigenschaft nicht unterstützt wird. Wenn während des Festlegens von Eigenschaften der Schreibvorgang die Konsistenzprüfungen nicht besteht, wird eine Recoverable exception ausgelöst.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Beschreibung HMISoftware (Seite 676)

5.14 Funktionen der Versionskontrollschnittstelle (VCI)

5.14.1 Auf die VCI-Systemgruppe im Projekt zugreifen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können in TIA Portal Openness auf die VCI-Systemgruppe zugreifen, indem Sie aus dem Projekt ein VersionControlInterface als Service aufrufen. Das VersionControlInterface ist ein Service und gibt ein Objekt zurück, das nicht Null ist.

Programmcode

Ändern Sie folgenden Programmcode, um die Workspace-Systemgruppe aus dem VersionControlInterface abzurufen:

```
public void function01()
{
    //Accessing VCI system group in project
    Siemens.Engineering.Project project = tiaPortal.Projects[0];
    Siemens.Engineering.VersionControl.VersionControlInterface versionControlInterface =
    project.GetService<VersionControlInterface>();
    Siemens.Engineering.VersionControl.WorkspaceSystemGroup workspaceSystemGroup =
    versionControlInterface.WorkspaceGroup;
}
```

5.14.2 Benutzergruppen in VCI-Gruppe enumerieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um alle VCI-Workspace-Benutzergruppen in anderen VCI-Workspace-Gruppen zu enumerieren.

```
WorkspaceUserGroup workspaceGroup = ...;
WorkspaceUserGroupComposition userGroupComposition = workspaceGroup.Groups;
foreach (Siemens.Engineering.VersionControl.WorkspaceUserGroup workspaceUserGroup in
userGroupComposition)
{
    //...
}
```

Ändern Sie folgenden Programmcode, um auf einzelne Workspace-Benutzergruppen in anderen VCI-Workspace-Gruppen zuzugreifen:

```
WorkspaceUserGroup workspaceGroup = ...;
Siemens.Engineering.VersionControl.WorkspaceUserGroup workspaceUserGroup =
workspaceGroup.Groups.Find("Some Group Name");
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.14.3 VCI-Benutzergruppe in VCI-Gruppe erstellen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness eine Workspace-Benutzergruppe in einer Workspace-Gruppe erstellen:

Programmcode

Ändern Sie folgenden Programmcode, um eine Workspace-Benutzergruppe zu erstellen:

```
Project project = ...;
//Creating VCI user group in VCI group
VersionControlInterface versionControlInterface =
project.GetService<VersionControlInterface>();
WorkspaceSystemGroup workspaceGroupComposition = versionControlInterface.WorkspaceGroup;
WorkspaceUserGroup result =
workspaceGroupComposition.Groups.Create("NewWorkspaceUserGroup");
```

Hinweis

Um eine neue Workspace-Benutzergruppe anhand des Namens erstellen zu können, muss der angegebene Name gültig sein und darf nicht bereits in einer anderen Workspace-Benutzergruppe vorhanden sein.

5.14.4 Eigenschaften von VCI-Gruppen aktualisieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Eigenschaften von Workspace-Benutzergruppen aktualisieren.

Die folgende Eigenschaft ist zur Aktualisierung in einer Workspace-Benutzergruppe verfügbar:

Name der Eigenschaft	Rückgabotyp	Beschreibung	Zugriffsmöglichkeit
Name	System.String	Name der Workspace-Benutzergruppe	Lesen/Schreiben

Programmcode

Ändern Sie folgenden Programmcode, um den Namen der Workspace-Benutzergruppe zu aktualisieren:

```
//Updating VCI group properties  
var workspaceUserGroup = ...;  
workspaceUserGroup.Name = "New_Group_Name";
```

5.14.5 VCI-Benutzergruppe löschen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Workspace-Benutzergruppen löschen. Wenn Sie Workspace-Benutzergruppen löschen, werden alle anderen in der Workspace-Benutzergruppe enthaltenen Objekte ebenfalls gelöscht. Hierzu gehören auch andere Workspace-Benutzergruppen und VCI-Workspaces.

Programmcode

Ändern Sie folgenden Programmcode, um eine Workspace-Benutzergruppe zu löschen:

```
//Deleting VCI user group  
WorkspaceUserGroup workspaceUserGroup = ...;  
workspaceUserGroup.Delete();
```

5.14.6 VCI-Workspaces in VCI-Gruppe enumerieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um alle VCI-Workspaces in einer VCI-Gruppe zu enumerieren:

```
WorkspaceSystemGroup workspaceSystemGroup = ...;
WorkspaceComposition workspaceComposition = workspaceSystemGroup.Workspaces;
foreach (Siemens.Engineering.VersionControl.Workspace workspace in workspaceComposition)
{
    //...
}
```

Ändern Sie folgenden Programmcode, um einen bestimmten Workspace anhand des Namens zu finden:

```
WorkspaceUserGroup workspaceUserGroup = ...;
//Enumerating VCI workspaces in VCI group 02
Siemens.Engineering.VersionControl.Workspace workspace =
workspaceUserGroup.Workspaces.Find("SomeWorkspaceName");
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

5.14.7 VCI-Workspace in VCI-Gruppe erstellen**Voraussetzung**

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness einen Workspace in einer Workspace-Benutzergruppe erstellen.

Einen `Siemens.Engineering.VersionControl.Workspace` können Sie mit der folgenden Erstellungsaktion erstellen:

```
Siemens.Engineering.VersionControl.WorkspaceComposition.Create(string name)
```

Programmcode

Ändern Sie folgenden Programmcode, um einen Workspace zu erstellen:

```
Project project = ...;
WorkspaceUserGroup workspaceUserGroup = ...;
//Creating VCI workspace in VCI group
VersionControlInterface versionControlInterface =
project.GetService<VersionControlInterface>();
WorkspaceComposition workspaceComposition =
versionControlInterface.WorkspaceGroup.Workspaces;
Workspace result = workspaceComposition.Create("NewWorkspace");
```

Hinweis

Um einen neuen Workspace anhand des Namens erstellen zu können, muss der angegebene Name gültig sein und darf nicht bereits in einer anderen Workspace-Benutzergruppe vorhanden sein.

5.14.8 VCI-Workspace-Eigenschaften aktualisieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die Eigenschaften eines VCI-Workspace aktualisieren.

Die folgenden Eigenschaften sind zur Aktualisierung in einem Workspace verfügbar:

Name der Eigenschaft	Rückgabotyp	Beschreibung	Zugriffsmöglichkeit
Name	System.String	Der Name des Work-space.	Lesen/Schreiben
RootPath	System.IO.DirectoryInfo	Der konfigurierte Work-space-Pfad.	Lesen/Schreiben

Programmcode: Eigenschaft "Name" aktualisieren

Ändern Sie folgenden Programmcode, um die Eigenschaft "Name" im Workspace zu aktualisieren:

```
var workspace = ...;  
workspace.Name = "New_Workspace_Name";
```

Programmcode: Eigenschaft "Rootpath" aktualisieren

Ändern Sie folgenden Programmcode, um den Workspace-Pfad zu einer Verzeichnisangabe mit Stammverzeichnis konfigurieren: Wird dieser Wert auf null gesetzt, so wird dadurch die Konfiguration des Workspace-Stammpfads aufgehoben.

```
var workspace = ...;  
workspace.RootPath = new DirectoryInfo(@"D:\Project_WS");
```

Ändern Sie folgenden Programmcode, um die Konfiguration eines Workspace aufzuheben:

```
var workspace = ...;  
workspace.RootPath = ...;
```

5.14.9 VCI-Workspace löschen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness einen Workspace löschen. Beim Löschen eines Workspace werden alle Zuordnungen des Workspace ebenfalls gelöscht.

Programmcode

Ändern Sie folgenden Programmcode, um einen Workspace zu löschen:

```
//Deleting VCI workspace  
Workspace workspace = ...;  
workspace.Delete();
```

5.14.10 VCI-Workspace-Zuordnungen in VCI enumerieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Ändern Sie folgenden Programmcode, um alle VCI-Workspace-Zuordnungen in einem Workspace zu enumerieren:

```
var workspace = ...;
WorkspaceMappingComposition mappings = workspace.Mappings;
foreach (Siemens.Engineering.VersionControl.WorkspaceMapping workspaceMapping in mappings)
{
    //...
}
```

Ändern Sie folgenden Programmcode, um auf einzelne Workspace-Zuordnungen für das spezielle Engineering-Objekt zuzugreifen:

```
PlcBlock block = ...;
var workspace = ...;
IEngineeringObject versionControlSupportedEngineeringObject = block;
Siemens.Engineering.VersionControl.WorkspaceMapping workspaceMapping =
workspace.Mappings.Find(versionControlSupportedEngineeringObject);
```

Hinweis

Die Schnittstelle IEngineeringObject wird von IEngineeringCompositionObject und IEngineeringInstance abgeleitet. Weitere Informationen zu IEngineeringObject finden Sie unter Dynamisches Verhalten (Seite 109)

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

5.14.11 VCI-Workspace-Zuordnung in VCI-Workspace erstellen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness eine Workspace-Zuordnung in einem VCI-Workspace erstellen.

Eine `Siemens.Engineering.VersionControl.WorkspaceMapping` können Sie mit der folgenden Signatur einer Erstellungsaktion erstellen:

```
Siemens.Engineering.VersionControl.WorkspaceMappingComposition.Create(  
    string relativeWorkspacePath, IEngineeringObject  
    linkedProjectObject)
```

Programmcode

Ändern Sie folgenden Programmcode, um eine Workspace-Zuordnung zu erstellen:

```
//Creating VCI workspace mapping in VCI workspace  
var workspace = ...;  
var plcBlock = ...;  
var result = workspace.Mappings.Create(@"\TestCopy\Block_1.xml", plcBlock);
```

Die wiederherstellbare Ausnahme wird in folgenden Fällen ausgegeben:

- Das verknüpfte Objekt ist kein gültiges VCI-unterstütztes Objekt.
- Das verknüpfte Objekt ist bereits zugeordnet.
- Der relative Dateipfad enthält ungültige Dateizeichen.
- Der relative Dateipfad enthält die Navigation zum übergeordneten Verzeichnis.

5.14.12 VCI-Workspace-Zuordnungen aktualisieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit dem TIA Openness Portal die Eigenschaften einer VCI-Workspace-Zuordnung aktualisieren.

Die folgenden Eigenschaften sind bei einer Workspace-Zuordnung verfügbar:

Name der Eigenschaft	Rückgabebetyp	Beschreibung	Zugriffsmöglichkeit
RelativeWorkspacePath	System.String	Der relative Dateipfad zur Darstellung der verknüpften Objekte in der Quellcodeverwaltung.	Lesen/Schreiben
LinkedProjectObject	IEngineeringObject	Das mit VCI kompatible Planungsobjekt, das mit der Datei in der Quellcodeverwaltung verknüpft ist.	Lesen

Programmcode

Ändern Sie folgenden Programmcode, um den relativen Dateipfad der Workspace-Zuordnung zu aktualisieren:

```
//Updating VCI workspace mapping properties
var workspaceMapping = ...;
workspaceMapping.RelativeWorkspacePath = @"\"Test\NewBlock_1.xml";
```

5.14.13 VCI-Workspace-Zuordnung löschen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness eine VCI-Workspace-Zuordnung löschen.

Programmcode

Ändern Sie folgenden Programmcode, um eine VCI-Workspace-Zuordnung zu löschen:

```
//Deleting VCI workspace mapping
WorkspaceMapping workspaceMapping = ...;
workspaceMapping.Delete();
```

5.14.14 Workspace-Zuordnung synchronisieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness den Status einer Workspace-Zuordnung anzeigen.

Programmcode: Synchronisationsstatus einzelner Objekte aktualisieren

Um den Status einzelner Objekte anzuzeigen, zu aktualisieren oder zu synchronisieren, müssen Sie zunächst aus einer Workspace-Zuordnung den `Siemens.Engineering.VersionControl.WorkspaceMapping.IndividualObjectSynchronizationStatus` anfordern.

Es wird empfohlen, vor dem Aufruf von Aktionen des Synchronisationsdienstes für einzelne Objekte eine Prüfung auf null durchzuführen, da unklar ist, ob die bestehende Zuordnung die Synchronisation einzelner Objekte unterstützt.

```
var individualObjectSynchronizationStatus =
workspaceMapping.GetService<IndividualObjectSynchronizationStatus>();
if(individualObjectSynchronizationStatus != null);
{
    //GetStatus()...
    //UpdateStatus()..
    //Synchronize()..
}
```

Programmcode: Status abrufen

Sie können den Synchronisationsstatus eines einzelnen Objekts mit Hilfe der folgenden Aktion anzeigen:

`Siemens.Engineering.VersionControl.IndividualObjectSynchronizationStatus.GetStatus()`. Die Aktion gibt `Siemens.Engineering.VersionControl.IndividualObjectCompareResult` zurück.

Das Objekt `IndividualObjectCompareResult` informiert über den aktuellen Synchronisationsstatus.

```
public class IndividualObjectCompareResult
{
    CompareState CompareState { get; }
    IndividualObjectCompareDetails {get; }
}
```

5.14 Funktionen der Versionskontrollschnittstelle (VCI)

Mit der Merker-Enumeration `IndividualObjectCompareDetails` werden Sie darüber informiert, welcher Teil der Workspace-Zuordnung geändert wurde (oder auch, dass beide Teile geändert wurden). Dieser Wert ist `None`, wenn der

`CompareState Equal` lautet. Wenn der Wert für `CompareState` "ungleich" lautet, kann diese Enumeration den Wert `ProjectObjectChanged` oder `WorkspaceFileChanged` oder sowohl `ProjectObjectChanged` **als auch**

`WorkspaceFileChanged` haben.

```
[Flags]public enum IndividualObjectCompareDetails
{
    None = 0,
    ProjectObjectChanged = 1,
    WorkspaceFileChanged = 2
}
```

Die Enumeration `CompareState` informiert darüber, ob ein Objekt Änderungen aufweist.

```
public enum CompareState
{
    Equal,
    Unequal,
    WorkspaceFileMissing,
    Unknown
}
```

Ändern Sie folgenden Programmcode, um den Status einer Workspace-Zuordnung abzurufen:

```
var workspaceMapping = ...;
var individualObjectSynchronizationStatus =
workspaceMapping.GetService<IndividualObjectSynchronizationStatus>();
if(individualObjectSynchronizationStatus != null)
{
    var compareResult = individualObjectSynchronizationStatus.GetStatus();
    //when compareState is unequal
    {
        if(compareResult.CompareState == CompareState.Unequal)
        {
            if(compareResult.IndividualObjectCompareDetails ==
IndividualObjectCompareDetails.WorkspaceFileChanged)
            {
                //Workspace file has changed
            }
            elseif(compareResult.IndividualObjectCompareDetails ==
(IndividualObjectCompareDetails.ProjectObjectChanged |
IndividualObjectCompareDetails.WorkspaceFileChanged))
            {
                //Both project object and workspace file has changed
            }
        }
    }
}
```

Programmcode: Synchronisationsstatus aktualisieren

Sie können das System dazu zwingen, einen Echtzeitvergleich zwischen dem aktuell verknüpften Projektobjekt und der Workspace-Datei durchzuführen. Der Vergleich kann bewirken, dass der Synchronisationsstatus auf den aktuellen Systemzustand geändert wird, und kann außerdem dazu dienen, den Status einer vor Kurzem verknüpften (aber nicht synchronisierten) Zuordnung festzustellen, ohne dabei das vorhandene Projektobjekt oder die Workspace-Datei zu verändern.

Ändern Sie folgenden Programmcode, um den Status einer Workspace-Zuordnung zu aktualisieren:

```
var workspaceMapping = ...;
var individualObjectSynchronizationStatus =
workspaceMapping.GetService<IndividualObjectSynchronizationStatus>();
if(individualObjectSynchronizationStatus != Null)
{
individualObjectSynchronizationStatus.UpdateStatus();
}
```

Programmcode: Workspace synchronisieren

Sie können das System dazu zwingen, einen Echtzeitvergleich zwischen dem aktuell verknüpften Projektobjekt und der Workspace-Datei durchzuführen. Der Vergleich kann bewirken, dass der Synchronisationsstatus auf den aktuellen Systemzustand geändert wird, und kann außerdem dazu dienen, den Status einer vor Kurzem verknüpften (aber nicht synchronisierten) Zuordnung festzustellen, ohne dabei das vorhandene Projektobjekt oder die Workspace-Datei zu verändern.

Mit der Enumeration SynchronizationMode wird das System darüber informiert, in welche Richtung es die Synchronisation vornehmen soll.

```
public enum SynchronizationMode
{
ProjectToWorkspace,
WorkspaceToProject
}
```

Mit dieser Aktion wird das System angewiesen, die Synchronisation vom verknüpften Projektobjekt zum verknüpften Workspace-Dateiobjekt bzw. in umgekehrter Richtung durchzuführen.

Ändern Sie folgenden Programmcode, um einen Workspace zu synchronisieren:

```
var workspaceMapping = ...;
var individualObjectSynchronizationStatus =
workspaceMapping.GetService<IndividualObjectSynchronizationStatus>();
if(individualObjectSynchronizationStatus != null)
{
individualObjectSynchronizationStatus.Synchronize(SynchronizationMode.ProjectToWorkspace);
}
```

5.14.15 Auf den Synchronisationsstatus untergeordneter Objekte zugreifen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness den Synchronisationsstatus untergeordneter Objekte abrufen.

Methode

Methodenname	Beschreibung
GetService<ChildObjectsStateProvider>()	Gibt das Objekt ChildObjectsStateProvider-Dienst aus der Workspace-Zuordnung an dem Objekt zurück, das über mindestens ein VCI-aktiviertes untergeordnetes Objekt verfügt.

Das Objekt ChildObjectStateProvider bietet folgende Methode, um den Synchronisationsstatus untergeordneter Objekte anzuzeigen.

Methodenname	Beschreibung
GetStatus()	Gibt das Objekt SynchronizationResult zurück, das den aktuellen Synchronisationsstatus untergeordneter Objekte darstellt.

ENUM

Das Objekt SynchronizationResult bietet folgende Enumeration, um mögliche Änderungen eines Objekts anzuzeigen.

Enum	Wert	Beschreibung
MappingState	Equal	Alle untergeordneten Objekte sind zugeordnet und synchronisiert.
	Unequal	Ein untergeordnetes Objekt eines hierarchischen Objekts ist nicht synchronisiert.

Programmcode

```
var workspaceMapping = ...;
//To get the ChildObjectStateProvider object from the workspace mapping that contains the
child of hierarchical objects.
var childObjectsStateProvider = workspaceMapping.GetService<ChildObjectsStateProvider>();
//To perform null check before invoking the actions on child object synchronization service
as the mapping may or may not support child objects synchronization.
if(childObjectsStateProvider != null)
{
//The GetStatus() returns all the child objects are mapped and are in sync.
//The SynchronizationResult returns the mapping status as Equal.
var synchronizationResult= childObjectsStateProvider.GetStatus();
//When MappingState is unequal
if(synchronizationResult.MappingState == MappingState.Unequal)
{
//Access to an individual child object and retrieve the status
}
}
```

5.14.16 VCI-Workspace mit Workspace-Sprache erstellen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness über die API `Workspace.Create()` im Version Control Interface (VCI) wie folgt einen Workspace erstellen:

- Erstellung eines Workspace mit der Workspace-Sprache, die als TIA Portal-Oberflächensprache voreingestellt ist, und Festlegung des Stammpfads als Parameter in der `API Workspace.Create()` .
- Erstellung eines Workspace mit Workspace-Sprache durch Zuweisung von Workspace-Sprache und Stammpfad als Parameter in der `API Workspace.Create()` .

Hinweis

In TIA Portal Openness V18 und Vorgängerversionen können Sie einen Workspace mit Workspace-Namen erstellen. Die Sprache wird implizit auf die voreingestellte TIA Portal-Oberflächensprache festgelegt und der Stammpfad muss nach dem Aufruf der API `Workspace.Create()` explizit gesetzt werden.

Parameter

Workspace.Create() akzeptiert zur Erstellung eines Workspace mit Workspace-Sprache im VCI folgende Parameter:

Parametername	DataType	Beschreibung
Name	String	Gibt den Namen des zu erstellenden Workspace an.
RootPath	DirectoryInfo	Gibt den Pfad zur Konfiguration des Workspace an.
WorkspaceLanguage	System.Globalization.CultureInfo	Gibt die konfigurierte Workspace-Sprache an.

Methode

Workspace.Create() erstellt den Workspace im VCI:

Methodenname	Beschreibung
Workspace.Create(string name, DirectoryInfo rootPath, CultureInfo workspaceLanguage)	Erstellt einen neuen Workspace mit dem angegebenen Namen, Stammpfad und Workspace-Sprachkultur.
Workspace.Create(string name, DirectoryInfo rootPath)	Erstellt einen neuen Workspace mit dem angegebenen Namen und dem Stammpfad. Die voreingestellte TIA Portal-Oberflächensprache wird der Workspace-Sprache zugewiesen.
Workspace.Create(string name)	Erstellt einen neuen Workspace mit dem angegebenen Namen. Die voreingestellte TIA Portal-Oberflächensprache wird der Workspace-Sprache zugewiesen; der Stammpfad ist explizit vom Benutzer festzulegen.

Programmcode

Erstellung eines Workspace mit angegebenem Workspace-Namen und der Sprache, die als TIA Portal-Oberflächensprache voreingestellt ist. Der Stammpfad ist explizit vom Benutzer festzulegen. Ändern Sie dazu folgenden Programmcode:

```
using Siemens.Engineering.VersionControlInterface;
private void createWorkspace()
{
    Workspace workspace = WorkspaceGroup.Workspaces.Create("workspace1");
    workspace.RootPath = new DirectoryInfo(Directory.GetCurrentDirectory());
}
```

Ändern Sie zur Erstellung eines Workspace mit der Workspace-Sprache, die als TIA Portal-Oberflächensprache voreingestellt ist, und Festlegung des Stammpfads als Parameter in der API `workspace.create()` folgenden Programmcode:

```
using Siemens.Engineering.VersionControl;
private void createWorkspace()
{
    WorkspaceComposition workspace = WorkspaceGroup.Workspaces.Create("workspace1", new
    DirectoryInfo(Directory.GetCurrentDirectory()));
}
```

Ändern Sie zur Erstellung eines Workspace mit Workspace-Sprache über deren Zuweisung und für den Stammpfad als Parameter in `workspace.create()` folgenden Programmcode:

```
using Siemens.Engineering.VersionControl;
private void createWorkspaceLanguage()
{
    Workspace workspace = WorkspaceGroup.Workspaces.Create("workspace1", new
    DirectoryInfo(Directory.GetCurrentDirectory()), new CultureInfo("de-DE"));
}
```

Hinweis

In den folgenden Fällen wird eine "EngineeringTargetException"-Ausnahme ausgelöst:

- Ungültige Workspace-Sprache
 - Festlegung der Workspace-Sprache in der API `workspace.create()` bei bereits im Stammpfad vorhandener Workspace-Sprache.
-

5.14.17 Initialisierung des VCI-Objektstatus

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie den Status des VCI-Objekts auf die abgebildete Datei initialisieren. Die API kann nur verwendet werden, wenn Sie sicher sind, dass die Dateien und Projektobjekte nicht verändert werden.

Die API ist über den Dienst `IndividualObjectSynchronizationStatus` als Teil der Arbeitsbereichszuordnung verfügbar und gibt als erfolgreiches Ergebnis für den VCI-Status grün zurück.

5.14 Funktionen der Versionskontrollschnittstelle (VCI)

Die Ausnahme `EngineeringTargetInvocationException` wird ausgelöst, wenn der VCI-Status unbekannt ist, zum Beispiel n ("?" in UI).

Methode

Methodenname	Beschreibung
<code>public void InitializeStatus()</code>	Initialisiert den Status des VCI-Objekts und die abgebildete Arbeitsbereichsdatei und stellt den Status auf Grün/Gleich.

Programmcode

```
var workspaceMapping=...;
var individualObjectSynchronizationStatus =
workspaceMapping.GetService<IndividualObjectSynchronizationStatus>();
if (individualObjectSynchronizationStatus != null)
{
    var currentStatus = individualObjectSynchronizationStatus.GetStatus().CompareState;
    if (currentStatus.Equals(CompareState.Unknown))
    {
        individualObjectSynchronizationStatus.InitializeStatus();
    }
}
```

5.14.18 Importieren von Simatic ML mit inaktiver Kultur

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit den Einstellungsoptionen für den VCI-Import können Sie die inaktiven Projektsprachen (inaktive Kulturen) im Projekt automatisch aktivieren, wenn ein VCI-Import über eine Simatic ML-Datei mit einer im TIA-Projekt nicht aktivierten Sprachkultur ausgelöst wird.

Sie können auf die Einstellung der Benutzeroberfläche im TIA Portal auch mithilfe von Openness über die Klasse `TiaPortalSettingsFolder` zugreifen. Die Klasse `TiaPortalSettingsFolder` ist über die Eigenschaft `Settings` der Klasse `TiaPortal` zugänglich.

Eigenschaft

Die VCI-Einstellungen werden über die Klasse TiaPortalSettingsFolder bereitgestellt. Die TiaPortalSettingsFolder-Instanz hat den Namen VCI.

Namen einstellen	Datentyp einstellen	Beschreibung	Zugriff
SimaticMLImportOptions	System.Int32	Aktiviert die inaktive Kultur/Sprache beim SimaticMLimport oder blockiert den Import, wenn die Kultur/Sprache nicht aktiv ist.	R/W

SimaticMLImportOptions ist eine Zahlen-Enum, die im Namensraum VersionControl mit den folgenden Werten definiert ist:

Enum-Optionen	Wert	Beschreibung
SimaticMLImportOptions.ActivateInactiveCultures	1	Aktiviert die Kulturen, wenn die Kulturen im TIA Portal-Projekt inaktiv sind
SimaticMLImportOptions.DoNotActivateInactiveCultures	0	Aktiviert die Kulturen nicht, wenn die Kulturen im TIA Portal-Projekt inaktiv sind

Programmcode

...

```
TiaPortalSettingsFolder vciSettingsFolder = TiaPortal.SettingsFolders.Find("VCI");
TiaPortalSetting inactiveCultureSetting =
vciSettingsFolder.Settings.Find("SimaticMLImportOptions");
// For Activating the inactiveCultures:
inactiveCultureSetting.Value = SimaticMLImportOptions.ActivateInactiveCultures;
//
For Not Activating the inactiveCultures:
inactiveCultureSetting.Value = SimaticMLImportOptions.DoNotActivateInactiveCultures;
```

5.15 Funktionsunterstützung für Mehrbenutzerbetrieb

5.15.1 Verbindung zum Projektserver erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit `Create()` eine neue Projektserververbindung in den Einstellungen von TIA Portal anlegen. `Create()` gibt den `ProjectServer` zurück und akzeptiert die folgenden Parameter:

Parameter	Datentyp	Beschreibung
aliasName	string	Gibt den Aliasnamen zum Erstellen der Projektserververbindung zurück
protocol	Protocol	Gibt den Protokolltyp zum Erstellen der Projektserververbindung zurück
hostname	string	Gibt den Hostnamen zum Erstellen der Projektserververbindung zurück
port	int	Gibt die Portnummer zum Erstellen der Projektserververbindung zurück

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
Protocol protocol = Protocol.Https;
string hostName = "localhost";
int port = 16000;
ProjectServer projectServer = tiaPortal.ProjectServers.Create(aliasName, protocol,
hostName, port);
```

Siehe auch

Projekt öffnen (Seite 131)

5.15.2 Projektserververbindung bearbeiten

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit `SetProtocol()`, `SetHostName()` und `SetPort()` in TIA Portal Openness eine Projektserververbindung in den Einstellungen von TIA Portal bearbeiten. Mit der API bearbeitbar sind die Einträge für Protokoll, Hostname und Port-Parameter.

Hinweis

Muss ein Aliasname geändert werden, muss der Eintrag mit der Projektserververbindung gelöscht und eine neue Verbindung angelegt werden.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
projectServer.SetProtocol(Protocol.Http);
projectServer.SetHostName("RemoteMachineHostName");
projectServer.SetPort(17000);
```

Siehe auch

Projekt öffnen (Seite 131)

5.15.3 Projektserververbindung löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit `DeleteConnection()` können Sie die Projektserververbindung in den Einstellungen von TIA Portal löschen.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();  
string aliasName = "ProjectServer1";  
ProjectServer projectServer =  
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));  
projectServer.DeleteConnection();
```

Siehe auch

Projekt öffnen (Seite 131)

5.15.4 Herstellen der Verbindung zum Projektserver aus den TIA Portal-Einstellungen heraus

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können TIA Portal Openness für den Zugriff auf Verbindungen zum Projektserver aus den TIA Portal-Einstellungen heraus verwenden. Die Openness API positioniert die Verbindungen zum Projektserver als direkter Navigator des TIA Portal. Verbindungen zum Projektserver werden in der Zusammensetzung "ProjectServers" des TIA Portal gesammelt angeordnet.

Jede "ProjectServer"-Instanz repräsentiert eine Konfiguration einer Projektserververbindung. Bei Verwendung der "ProjectServer"-Instanz, die aus der Zusammensetzung "ProjectServers" abgerufen wurde, können Sie bequem mit dem TIA-Projektserver arbeiten.

Jede Funktionalität, die Interaktionen mit einem Multiuser-Server ohne Öffnen einer lokalen Sitzung einbezieht, kann mit Hilfe dieses Navigators aufgerufen werden und "ProjectServers" sind jederzeit verfügbar.

Der lokale Windows-Benutzer wird bei der Authentifizierung mit dem Multiuser-Server herangezogen.

Programmcode

Sie können den nachfolgenden Beispielcode für den Zugriff auf Projektserver-relevante Eigenschaften verwenden:

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
// The project server has ServerName property, referred to the "alias" assigned as the name
for the project server is used as an
identifier.
// For example, if full URL of server is "net.tcp://localhost:9876/"
// the projectServer.Host returns "net.tcp://localhost/" and
// projectServer.Port returns 9876
string ServerName = projectServer.ServerName;
string host = projectServer.Host;
int portNumber = projectServer.Port;
```

5.15.5 Ein Projekt im Server hinzufügen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit `AddProjectToServer()` ein Einzelbenutzerprojekt einem Multiuser-Server hinzufügen. `AddProjectToServer()` akzeptiert die folgenden Parameter für das Projekt. Ist das Projekt, das hinzugefügt wird, bereits geöffnet, wird es jetzt geschlossen.

Parameter	Datentyp	Beschreibung
projectFileInfo	FileInfo	Gibt die Informationen für den Serverpfad an

Hinweis

Für alle APIs wird der lokale Windows-Benutzer, der die Openness-Anwendung / das Openness-Skript ausführt, zur Authentifizierung beim Multiuser-Server herangezogen. Es ist wichtig, dass der Benutzer dem Multiuser-Server mit der erforderlichen Rolle für den beabsichtigten Vorgang hinzugefügt wird. Wenn der Benutzer nicht authentifiziert wird, wird eine Ausnahme ausgelöst.

Programmcode

Ändern Sie folgenden Programmcode, um ein Einzelbenutzerprojekt in einem Multiuser-Server hinzuzufügen:

```
TiaPortal tiaPortal = new TiaPortal();  
string aliasName = "ProjectServer1";  
FileInfo projectFileInfo = new FileInfo("C:\\Projects\\Project1\\Project1.apl7");  
ProjectServer projectServer =  
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));  
projectServer.AddProjectToServer(projectFileInfo);
```

5.15.6 Serverprojekte abrufen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können mit `GetServerProjects()` eine Liste mit Informationen zu den verfügbaren Serverprojekten auf dem Projektserver abrufen. `GetServerProjects()` gibt `IEnumerable<ServerProjectInfo>` zurück.

Sie können diese Informationen für Vorgänge wie die folgenden verwenden:

- Informationen zu lokalen Sitzungen für ein bestimmtes Serverprojekt abrufen
- Eine lokale Sitzung für ein bestimmtes Serverprojekt erstellen

Programmcode

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;

using Siemens.Engineering.SW.Tags;

using Siemens.Engineering.SW.Types;

using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
using Siemens.Engineering.Multiuser;
namespace GettingProjectServer
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            TiaPortal tiaPortal = new TiaPortal();
            string aliasName = "ProjectServer1";
            ProjectServer projectServer =
            tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
            IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects()
        }
    }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.15.7 Verfügbare lokale Sitzungen abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer Anwendung TIA Portal Openness ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Sie können mit `GetLocalSessions()` eine Liste mit Informationen zu verfügbaren Sitzungen für ein bestimmtes, vom aktuellen Benutzer angelegtes Serverprojekt auf dem aktuellen Rechner abrufen.

Mit diesen Informationen können Sie weitere Aktionen wie das Öffnen oder Löschen der lokalen Sitzung ausführen.

`GetLocalSessions()` gibt `IEnumerable<ServerProjectInfo>` zurück und hat den Parameter `serverProjectInfo`, dessen Datentyp `ServerProjectInfo` lautet.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
IEnumerable<LocalSessionInfo> localSessionsInfo =
projectServer.GetLocalSessions(serverProjectInfo[0]);
```

Siehe auch

Projekt öffnen (Seite 131)

5.15.8 Eine lokale Sitzung erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt wird geöffnet
Siehe Projekt öffnen (Seite 131)

Anwendung

Sie können mit `CreateLocalSession()` eine lokale Sitzung für ein bestimmtes Serverprojekt erstellen und das Objekt `LocalSessionInfo` zurückgeben.

`CreateLocalSession()` akzeptiert die folgenden Parameter zum Erstellen einer lokalen Sitzung mit TIA Portal Openness:

Parameter	Datentyp	Beschreibung	Hinweis
serverProjectInfo	ServerProjectInfo	Gibt die Liste der verfügbaren Projekte auf Serverprojekten an	
localSessionName	String	Gibt den Namen der lokalen Sitzung an	
localSessionPath	DirectoryInfo	Gibt den Verzeichnispfad an, in dem die lokale Sitzung erstellt wird	
mode	SessionCreationMode	SessionCreationMode akzeptiert die folgenden Arten von ENUM-Werten: - SessionCreationMode.Multiuser - SessionCreationMode.Exclusive	Das Serverprojekt wird automatisch gesperrt, wenn die Sitzung als exklusive lokale Sitzung angelegt wird.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo =
projectServer.CreateLocalSession(serverProjectInfo.First(), localSessionName,
directoryInfo,
SessionCreationMode.Multiuser);
LocalSessionInfo localSessionInfo =
projectServer.CreateLocalSession(serverProjectInfo.First(), localSessionName,
directoryInfo,
SessionCreationMode.Exclusive);
```

5.15.9 Eine lokale Sitzung speichern

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit `Save()` eine Sitzung speichern. `Save()` gibt den leeren Typ zurück und speichert die Sitzung nur, wenn es sich um eine Sitzung von Typ 'Engineering' oder 'exklusiv' handelt. Andernfalls wird die `MultiuserException` ausgelöst, wenn es sich bei der Sitzung um ein Serverprojekt handelt.

Programmcode

```
// ...
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo =
projectServer.CreateLocalSession(serverProjectInfo.First(), localSessionName,
directoryInfo,
SessionCreationMode.Exclusive);
LocalSession localSession = tiaPortal.LocalSessions.Open(localSessionInfo.ProjectFileInfo);
//Edit project
localSession.Save();
// ...
```

5.15.10 Lokale Sitzung auf dem Server löschen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit `DeleteLocalSessionOnServer()` können Sie eine bestimmte lokale Sitzung löschen. Mit `DeleteLocalSessionOnServer()` wird die Sitzung nur vom Multiuser-Server gelöscht. Die Sitzungsdateien auf dem lokalen Laufwerk werden nicht gelöscht. Die Sperre des Serverprojekts wird automatisch entfernt, wenn es sich bei der angegebenen Sitzung um eine exklusive Sitzung handelt.

DeleteLocalSessionOnServer() akzeptiert die folgenden Parameter zum Löschen einer lokalen Sitzung:

Parameter	Datentyp	Beschreibung
serverProjectInfo	ServerProjectInfo	Gibt die Liste der verfügbaren Projekte auf Serverprojekten an
localsessioninfo	LocalSessionInfo	Gibt die lokalen Sitzungsinformationen für das Projekt an

Programmcode

```
//...
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo =
projectServer.CreateLocalSession(serverProjectInfo.First(), localSessionName,
directoryInfo);
string serverAlias = serverProjectInfo[0].ServerAlias;
string projectName = serverProjectInfo[0].ProjectName;
int sessionId = localSessionInfo.SessionId;
projectServer.DeleteLocalSessionFromServer(serverProjectInfo, localSessionInfo);
//...
```

5.15.11 Lokale/exklusive Sitzung öffnen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit Open() eine lokale Multiuser-Sitzung oder eine exklusive lokale Sitzung für ein bestimmtes Serverprojekt öffnen. Open() gibt ein Objekt LocalSession vom Typ 'Engineering-Sitzung' oder 'lokale Sitzung' zurück und hat den Parameter localSessionPath, dessen Datentyp FileInfo lautet.

Das Serverprojekt wird für eine als exklusive Sitzung geöffnete Sitzung automatisch gesperrt.

Programmcode

Ändern Sie folgenden Programmcode, um eine Engineering-Sitzung zu öffnen:

```
:
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo = projectServer.CreateLocalSession(serverProjectInfo[0],
localSessionName, directoryInfo);
LocalSession multiuserLocalSession =
tiaPortal.LocalSessions.Open(localSessionInfo.ProjectFileInfo);
:
```

Ändern Sie folgenden Programmcode, um eine exklusive Sitzung zu öffnen:

```
:
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo = projectServer.CreateLocalSession(serverProjectInfo[0],
localSessionName, directoryInfo, SessionCreationMode.Exclusive);
LocalSession exclusiveSession =
tiaPortal.LocalSessions.Open(localSessionInfo.ProjectFileInfo);
:
```

Siehe auch

Projekt öffnen (Seite 131)

5.15.12 Serverprojekt öffnen**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit `OpenServerProject()` ein Serverprojekt für eine bestimmte lokale Sitzung öffnen. `OpenServerProject()` gibt das Objekt `LocalSession` vom Typ 'Serverprojekt' zurück und hat den Parameter `localSessionPath`, dessen Datentyp 'FileInfo' lautet.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo =
projectServer.CreateLocalSession(serverProjectInfo.First(), localSessionName,
directoryInfo,
SessionCreationMode.Exclusive);
LocalSession serverProject =
tiaPortal.LocalSessions.OpenServerProject(localSessionInfo.ProjectFileInfo);
```

Siehe auch

Projekt öffnen (Seite 131)

5.15.13 Markierte Objekte prüfen

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Informationen darüber bereitstellen, ob ein bestimmtes `EngineeringObject` markiert werden kann. TIA Portal Openness stellt auch markierungsbezogene Informationen bereit wie:

- ob das Objekt vom aktuellen Benutzer markiert ist
- ob das Objekt von anderen Benutzern markiert ist
- ob das Objekt aktuell ist

Sie können mit TIA Portal Openness ferner den Markierungsstatus eines verfügbaren `EngineeringObject` abrufen.

5.15 Funktionsunterstützung für Mehrbenutzerbetrieb

Auf die folgenden Methoden- und Parameternamen kann mit TIA Portal Openness zugegriffen werden:

Rückgabebetyp	Methodenname	Parameter
MarkStateInfo	GetMarkStateInfo	IEngineeringObject engineeringObject

Programmcode

```
//Open a local session or a server project
//Add a block or edit an block of type IEngineeringObject
MarkStateInfo markStateInfo = localSession.MarkingService.GetMarkStateInfo(muBlock);
if (markStateInfo.IsMarkable)
{
    // The object is markable
    // Now we can check the markstate for more information
    if (markStateInfo.MarkState.HasFlag(MarkState.IsUptoDate))
    {
        // You have the latest object
    }
    if (markStateInfo.MarkState.HasFlag(MarkState.IsMarkedByMe))
    {
        // You have the marked the object
    }
    if (markStateInfo.MarkState.HasFlag(MarkState.IsMarkedByOthers))
    {
        // Some other user has marked the object
    }
    //Example for Checkin without conflicts
    if (markStateInfo.MarkState.HasFlag(MarkState.IsUptoDate) &&
        markStateInfo.MarkState.HasFlag(MarkState.IsMarkedByMe) && !
        markStateInfo.MarkState.HasFlag(MarkState.IsMarkedByOthers))
    {
        // You have the latest object, the object is marked by me and no one else has marked the
        object
    }
}
else
{
    // The object is not markable. Open server project and make changes, then commit.
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.15.14 Eine lokale Sitzung schließen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit `Close()` ein Serverprojekt, eine Multiuser- und eine exklusive Sitzung schließen. Durch `Close()` wird ein leerer Typ zurückgegeben und alle nicht gespeicherten Änderungen gehen verloren.

Programmcode

Ändern Sie folgenden Programmcode, um das Serverprojekt zu schließen:

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo = projectServer.CreateLocalSession(serverProjectInfo[0],
localSessionName, directoryInfo);
LocalSession serverProject =
tiaPortal.LocalSessions.OpenServerProject(localSessionInfo.ProjectFileInfo);
serverProject.Close();
```

Ändern Sie folgenden Programmcode, um eine exklusive Sitzung zu schließen:

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "serverAliasName";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "ProjectServer1";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo = projectServer.CreateLocalSession(serverProjectInfo[0],
localSessionName, directoryInfo, SessionCreationMode.Exclusive);
LocalSession exclusiveSession =
tiaPortal.LocalSessions.Open(localSessionInfo.ProjectFileInfo);
exclusiveSession.Close();
```

Ändern Sie folgenden Programmcode, um eine Engineering-Sitzung zu schließen:

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo = projectServer.CreateLocalSession(serverProjectInfo[0],
localSessionName, directoryInfo);
LocalSession multiuserLocalSession =
tiaPortal.LocalSessions.Open(localSessionInfo.ProjectFileInfo);
multiuserLocalSession.Close();
```

5.15.15 Änderung des Serverprojekts übernehmen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit `CloseAndCommit()` die auf dem Serverprojekt und in der exklusiven Sitzung vorgenommenen Änderungen speichern. `CloseAndCommit()` gibt den Typ `integer` zurück und hat den Parameter `Comment` vom Typ `string`.

Programmcode

Ändern Sie das folgende Programm, um Änderungen im Serverprojekt zu speichern:

```
:
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo = projectServer.CreateLocalSession(serverProjectInfo[0],
localSessionName, directoryInfo);
LocalSession serverProject =
tiaPortal.LocalSessions.OpenServerProject(localSessionInfo.ProjectFileInfo);
int revisionCreated = serverProject.CloseAndCommit(comment: "Comment");
:
```

Ändern Sie folgenden Programmcode, um Änderungen an einer exklusiven Sitzung zu speichern:

```
:
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo = projectServer.CreateLocalSession(serverProjectInfo[0],
localSessionName, directoryInfo, SessionCreationMode.ExclusiveSession);
LocalSession exclusiveSession =
tiaPortal.LocalSessions.Open(localSessionInfo.ProjectFileInfo);
int revisionCreated = exclusiveSession.CloseAndCommit(comment: "Comment");
:
```

5.15.16 LockStateProvider abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit `GetLockStateProvider()` Informationen über die Sperre eines Serverprojekts und den Eigentümer der Sperre abrufen. `GetLockStateProvider()` gibt den Typ `LockStateProvider` zurück und hat den Parameter `serverProjectInfo`, dessen Datentyp `ServerProjectInfo` lautet.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
LockStateProvider lockStateProvider =
projectServer.GetLockStateProvider(serverProjectInfo[0]);
```

5.15.17 Gesperrtes Projekt prüfen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit Hilfe von `IsProjectLocked()` prüfen, ob ein Serverprojekt gesperrt oder nicht gesperrt ist. `IsProjectLocked()` gibt den Wert `bool` zurück und ruft seinen Echtzeitwert für das Projekt durch Kontaktieren des Servers ab.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer = tiaPortal.ProjectServers.First(a =>
a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
LockstateProvider lockstateProvider =
projectServer.GetLockstateProvider(serverProjectInfo[0]);
bool isServerProjectLocked = lockstateProvider.IsProjectLocked();
```

5.15.18 Eigentümer der Sperre abrufen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit `GetLockOwner()` Informationen über den Eigentümer der Sperre eines Serverprojekts abrufen. `GetLockOwner()` gibt `string` zurück und gibt eine leere Zeichenkette (`String`) zurück, wenn das Projekt nicht gesperrt ist.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer =
tiaPortal.ProjectServers.First(a=>a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
LockStateProvider lockStateProvider =
projectServer.GetLockStateProvider(serverProjectInfo[0]);
string lockOwner= lockStateProvider.GetLockOwner();
```

Siehe auch

Projekt öffnen (Seite 131)

5.15.19 Aktualität der Sitzung prüfen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit Hilfe von `IsUptoDate()` prüfen, ob eine Sitzung für mehrere Benutzer aktuell oder nicht aktuell ist. `IsUptoDate()` gibt einen booleschen Wert zurück.

Programmcode

```
TiaPortal tiaPortal = new TiaPortal();
string aliasName = "ProjectServer1";
ProjectServer projectServer = tiaPortal.ProjectServers.First(a =>
a.ServerName.Equals(aliasName));
IEnumerable<ServerProjectInfo> serverProjectInfo = projectServer.GetServerProjects();
string localSessionName = "testLocalSessionName";
DirectoryInfo directoryInfo = new DirectoryInfo("C:\\Sessions");
LocalSessionInfo localSessionInfo =
projectServer.CreateLocalSession(serverProjectInfo.First(), localSessionName,
directoryInfo,
SessionCreationMode.Exclusive);
LocalSession multiuserLocalSession =
tiaPortal.LocalSessions.Open(localSessionInfo.ProjectFileInfo);
bool isUptoDate = multiuserLocalSession.IsUptoDate();
```

Hinweis

Im obigen Programmcode kann der sessionmode für den Projektserver entweder "multiuser" oder "exclusive" lauten.

5.15.20 Markierungen aus lokaler Sitzung abrufen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet
Siehe Projekt öffnen (Seite 131)

Verwendung

Mit TIA Portal Openness können Sie alle Markierungen (Konflikt, kein Konflikt) einer lokalen Sitzung abrufen.

Sie können mit TIA Portal Openness die Online-/Echtzeitinformationen vom Multiuser-Server abrufen.

Auf die folgende Methode kann mit TIA Portal Openness zugegriffen werden:

Methodenname	Rückgabetyt
GetMarkings	Markings

Auf den folgenden Eigenschaftennamen kann mit TIA Portal Openness zugegriffen werden:

Benutzerdefinierter Typ	Name der Eigenschaft	Rückgabotyp
Markings	AllMarkings	MarkingComposition
Markings	ConflictedMarkings	MarkingComposition
Marking	MarkedObject	IEngineeringObject
Marking	MarkState	MarkState

Die TIA Portal Openness API gibt keine Markierungen zu gelöschten Objekten zurück, da das Openness-Framework gelöschte Objekte nicht darstellen kann.

Hinweis

Markierungen mit Konflikten sind solche, die von Ihnen oder anderen markiert oder von Ihnen markiert und veraltet oder von Ihnen und/oder von anderen markiert und veraltet sind.

Programmcode

```
//Open a local session or a server project
//All the non deleted object markings from the session will be retrieved.
Markings markings = localSession.MarkingService.GetMarkings();
MarkingComposition allMarkings = markings.AllMarkings;
foreach (Marking marking in allMarkings)
{
    IEngineeringObject markedObject = marking.MarkedObject;
    MarkState markState = marking.MarkState;
}
//All the non deleted marked objects which are conflicted.
MarkingComposition conflictedMarkings = markings.ConflictedMarkings;
```

5.16 Funktionen für den Zugriff auf Daten der Benutzerverwaltung

5.16.1 Funktionen für den Zugriff auf Benutzer, Rollen und Funktionsrechte

5.16.1.1 Projekt schützen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness das TIA Portal-Projekt schützen. Das Projekt wird von der API geschützt, die auch einen standardmäßigen Admin erstellt.

Das TIA Portal schützt das Projekt unter folgenden Bedingungen nicht:

- Wenn ein zuvor bereits geschütztes Projekt geschützt werden soll
- Wenn keine Meta-Konfiguration für mindestens ein UMAC-Gerät im Projekt existiert
- Wenn die Validierung der bereitgestellten Parameter fehlschlägt

Die `EngineeringTargetInvocationException` wird ausgelöst, wenn durch `protectProject()` kein Aufruf erfolgt.

Programmcode

```
private static SecureString GetSecureString(string value)
{
    SecureString secureStr = new SecureString();
    if (!string.IsNullOrEmpty(value))
    {
        for (int i = 0; i < value.Length; i++)
        {
            secureStr.AppendChar(value[i]);
        }
    }
    return secureStr;
}

private static void Main(string[] args)
{
    TiaPortal tiaPortal = new TiaPortal();
    Project tiaProject = tiaPortal.Projects[0];
    tiaProject.ProtectProject("AdminUser", GetSecureString("Admin123")); // see GetSecureString
    function
}
```

Siehe auch

- Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen (Seite 131)

5.16.1.2 Engineering-Funktionsrechte

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness die folgenden Informationen zu Engineering-Funktionsrechten abrufen:

- Alle Engineering-Funktionsrechte in einem TIA Portal-Projekt abrufen
- Ein bestimmtes Engineering-Funktionsrecht nach Bezeichner abrufen

Der Eigenschaftswert des Namens eines Engineering-Funktionsrechts beruht auf der Oberflächensprache. Der Eigenschaftswert des Bezeichners ist jedoch immer konstant, unabhängig von der Oberflächensprache.

Für "Find" sollte der Eigenschaftswert des API-Bezeichners verwendet werden.

Programmcode

```
private static SecureString GetSecureString(string value)
{
    SecureString secureStr = new SecureString();
    if (!string.IsNullOrEmpty(value))
    {
        for (int i = 0; i < value.Length; i++)
        {
            secureStr.AppendChar(value[i]);
        }
    }
    return secureStr;
}

Project tiaProject = ...;
tiaProject.protectProject("Admin123", ToSecureString("Admin123")); // see function
GetSecureString
UmacConfigurator UmacConfiguratorService = tiaProject.GetService<UmacConfigurator>();
//Retrieving Engineering function Rights. Returned list will be empty if called in non-
protected mode
var engineeringFunctionRights = UmacConfiguratorService.EngineeringFunctionRights;
string engineeringFunctionRightIdentifier;
string engineeringFunctionRightName;
//Iterating through Engineering function Rights
foreach((var engineeringFunctionRight in engineeringFunctionRights)
{
    engineeringFunctionRightName = engineeringFunctionRight.Name;
    engineeringFunctionRightIdentifier = engineeringFunctionRight.Identifier;
})
//Find an Engineering Function Right
var foundEngineeringFunctionRight =
    engineeringFunctionRights.Find("engineeringFunctionRightIdentifier");
```

Siehe auch

[Verbindung zum TIA Portal aufbauen \(Seite 84\)](#)

[Projekt öffnen \(Seite 131\)](#)

5.16.1.3 Funktionsrechte für Geräte

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Im TIA Portal gibt es zwei Arten von Funktionsrechten für Geräte:

- Funktionsrechte für Systemgeräte
- Funktionsrechte für benutzerdefinierte Geräte

Sie können mit TIA Portal Openness die verfügbaren Geräte-Funktionsrechte abrufen:

- Alle Geräte-Funktionsrechte für ein System und eine benutzerdefinierte Geräteinstanz abrufen
- Alle benutzerdefinierten Geräte-Funktionsrechte im Projekt abrufen
- Benutzerdefinierte Geräte-Funktionsrechte im Projekt anlegen
- Ein benutzerdefiniertes Geräte-Funktionsrecht nach Bezeichner abrufen

Hinweis

Beim Anlegen eines benutzerdefinierten Geräte-Funktionsrechts löst TIA Portal Openness eine `EngineeringTargetException` aus.

In den folgenden Fällen schlägt das Anlegen eines benutzerdefinierten Geräte-Funktionsrechts fehl:

- Die Obergrenze der Funktionsrechte für benutzerdefinierte Geräte ist erreicht und es können keine weiteren benutzerdefinierten Funktionsrechte erstellt werden
- Die Bestätigung bestimmter Parameter schlägt fehl (zum Beispiel Sonderzeichen oder Länge)

Für Systemgeräte-Funktionsrechte wird der Eigenschaftswert des Namens anhand der TIA Portal-Oberflächensprache definiert. Der Eigenschaftswert des Bezeichners ist unabhängig von der in der TIA Portal-Benutzeroberfläche eingestellten Sprache immer konstant.

Für "Find" sollte der Eigenschaftswert des API-Bezeichners verwendet werden.

Programmcode

```
Device device = ...;
Project project = ...;
var umacDevice = device.GetService<UmacDevice>();
// Get available system and custom device function rights in the project
var deviceFunctionRightAssociation = umacDevice.AvailableDeviceFunctionRights;
string name;
string identifier;
// Iterating through device function rights
foreach(var deviceFunctionRight in deviceFunctionRightAssociation)
{
    name = deviceFunctionRight.Name;
    identifier = deviceFunctionRight.Identifier;
}
// Retrieve the UmacConfiguratorService from project
UmacConfigurator umacConfiguratorService = project.GetService<UmacConfigurator>();
// Get all custom device function rights in the project
var CustomDeviceFunctionRightComposition =
umacConfiguratorService.AvailableCustomDeviceFunctionRights;
//create a custom device function right
var CustomDeviceFunctionRights =
customDeviceFunctionRightComposition.Create("RuntimeRight1", "General", "user defined
runtime right");
// Name and Identifier value for Custom Device Function Right will be the same.
// Iterating through custom device function rights
foreach(var customDeviceFunctionRight in CustomDeviceFunctionRights)
{
    name = customDeviceFunctionRight.Name;
    identifier = customDeviceFunctionRight.Identifier;
}
//Find a custom device function right
customDeviceFunctionRight = customDeviceFunctionRightComposition.Find("identifier");
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.16.1.4 Systemrollen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Bei der Verwendung von TIA Portal Openness können Sie die folgenden Aufgaben für benutzerdefinierte Rollen und Geräte-Funktionsrechte ausführen:

- Alle Systemrollen in einem Projekt abrufen
- Eine bestimmte Systemrolle in einem Projekt suchen
- Alle zugewiesenen Engineering-Rechte einer bestimmten Systemrolle abrufen
- Alle zugewiesenen Systemgeräte-Funktionsrechte für eine Systemrolle abrufen

Für Systemrollen wird der Wert der Namenseigenschaft anhand der Sprache der TIA Portal-Benutzeroberfläche definiert. Der Wert der Bezeichnereigenschaft ist unabhängig von der Sprache der Benutzeroberfläche des TIA Portals immer konstant.

Für "Find" sollte der Eigenschaftswert des API-Bezeichners verwendet werden.

Programmcode

```
Project project = ...;
Device device = ...;
UmacConfigurator umacConfiguratorService = project.GetService<UmacConfigurator>();
var umacDevice = device.GetService<UmacDevice>();
//Retrieve all system role
var systemRoles = umacConfiguratorService.SystemRoles;
//Iterate through the system roles
foreach (var sr in systemRoles)
{
    string systemRoleName = sr.Name;
}
// Find a system role
SystemRole systemRole = systemRoles.Find("SystemRole_01");
//Retrieve assigned System Device Function Rights from system role
IList<SystemDeviceFunctionRight> assignedSystemDeviceFunctionRights
=(IList<SystemDeviceFunctionRight>)systemRole.
GetAssignedSystemDeviceFunctionRights(umacDevice);
//Retrieve assigned Engineering Function Rights from system role
var engineeringFunctionRights = systemRole.AssignedEngineeringRights;
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.16.1.5 Benutzerdefinierte Rollen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Benutzerdefinierte Rollen sind Rollen, die vom Benutzer im Laufe der Projektlebensdauer dynamisch angelegt werden. Sie können mit TIA Portal Openness die folgenden Aufgaben mit benutzerdefinierten Rollen ausführen:

- Alle benutzerdefinierten Rollen in einem Projekt abrufen
- Eine benutzerdefinierte Rolle in einem Projekt anlegen
- Eine bestimmte benutzerdefinierte Rolle in einem Projekt suchen
- Attribute (Name und Kommentar) einer benutzerdefinierten Rolle bearbeiten
- Eine benutzerdefinierte Rolle in einem Projekt löschen
- Einer benutzerdefinierten Rolle ein EFR zuweisen/entziehen
- Einer benutzerdefinierten Rolle ein GFR über eine Geräteinstanz zuweisen/entziehen
- Einer benutzerdefinierten Rolle zugewiesene Engineering-Funktionsrechte abrufen
- Einer benutzerdefinierten Rolle zugewiesene Geräte-Funktionsrechte abrufen.

Der Eigenschaftswert des Bezeichners ist bei benutzerdefinierten Rollen immer konstant, unabhängig von der Oberflächensprache. Für "Find" sollte der Eigenschaftswert des API-Bezeichners verwendet werden.

Programmcode

```

TiaPortal tiaPortal = new TiaPortal();
Project tiaProject = tiaPortal.Projects[0];
UmacConfigurator UmacConfiguratorService = tiaProject.GetService<UmacConfigurator>();
// Retrieve custom roles in project
var customRoles = UmacConfiguratorService.CustomRoles;
// Create a custom role
CustomRole newCustomRole = customRoles.Create("Role_1", "some comment");
// Iterate through custom roles & edit it
foreach (CustomRole customRole in customRoles)
{
    if (newCustomRole.Name == "CustomRoleNew")
    {
        customRole.Comment = "CustomRoleNew is allowed to ...";
        //..
    }
}
// Find a custom role
newCustomRole = customRoles.Find("CustomRole1");
// Edit a custom role
newCustomRole.Comment = "some new comment";
// Retrieve Engineering Function Right. This list will be empty if called in non-protected
mode.
var engineeringFunctionRights = UmacConfiguratorService.EngineeringFunctionRights;
// Find an Engineering Function Right
EngineeringFunctionRight engineeringFunctionRight =
engineeringFunctionRights.Find("EFR_01");
// Assign a EFR to Custom Role
customRole.AssignedEngineeringRights.Add(engineeringFunctionRight);
// Retrieve assigned Engineering Function Rights on a custom role
var engineeringFunctionRights = customRole.AssignedEngineeringRights;
// Remove a EFR from Custom Role
customRole.AssignedEngineeringRights.Remove(engineeringFunctionRight);
// Get the umac device service from openness Device. If service is invoked on non-umac
device then value will be null
Device device = ...;
var umacDevice = device.GetService<UmacDevice>();
var deviceFunctionRightsAssociation = umacDevice.AvailableDeviceFunctionRights;
var customDeviceFunctionRight = deviceFunctionRightsAssociation.Find("User Right1");
var systemDeviceFunctionRight = deviceFunctionRightsAssociation.Find("SystemDFR_01");
// Assign a custom device function right to custom role
customRole.AssignDeviceFunctionRight(umacDevice, customDeviceFunctionRight);
//Assign a system device function right
customRole.AssignDeviceFunctionRight(umacDevice, systemDeviceFunctionRight);
//Retrieve assigned Device function Rights
var assignedDeviceFunctionRights = customRole.GetAssignedDeviceFunctionRights(umacDevice);
// Unassign a custom device function right from custom role
customRole.UnAssignDeviceFunctionRight(umacDevice, customDeviceFunctionRight);
//Unassign a system device function right
customRole.UnAssignDeviceFunctionRight(umacDevice, systemDeviceFunctionRight);
// Delete a Custom Role
customRole.Delete();

```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.16.1.6 Benutzer eines Projekts

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Die Benutzer eines Projekts sind Benutzer, die lokal für ein Projekt angelegt werden. Ist ein Projekt standardmäßig geschützt, wird ein Benutzer angelegt und erhält die ES-Administratorrolle.

Sie können mit TIA Portal Openness die folgenden Aufgaben für Benutzer eines Projekts ausführen:

- Alle Benutzer eines Projekts abrufen
- Einen Benutzer eines Projekts anlegen
- Einen Benutzer in einem Projekt anhand des Namens suchen
- Die zugewiesenen Rollen für den Benutzer eines Projekts abrufen
- Eine Rolle für den Benutzer eines Projekts hinzufügen/entfernen.
- Den Benutzer eines Projekts aktivieren/deaktivieren
- Gibt an, ob der Benutzer des Projekts aktiv ist

Programmcode

```
private static SecureString GetSecureString(string value)
{
    SecureString secureStr = new SecureString();
    if (!string.IsNullOrEmpty(value))
    {
        for (int i = 0; i < value.Length; i++)
        {
            secureStr.AppendChar(value[i]);
        }
    }
    return secureStr;
}

private static void protectProject()
{
    TiaPortal tiaPortal = new TiaPortal();
    Project tiaProject = tiaPortal.Projects[0];
    tiaProject.ProtectProject("AdminUser", GetSecureString("Admin123")); //see function
    GetSecureString
}

private static void projectUser()
{
    Project project = ...;
    UmacConfiguratorService = project.GetService<UmacConfigurator>();
    //Retrieve custom roles in project
    var customRoles = UmacConfiguratorService.CustomRoles;
    var customRole = customRoles.Find("Role_1");
    var systemRole = customRoles.Find("Engineering administrator");
    //Get project users
    var projectUsers = UmacConfiguratorService.ProjectUsers;
    // Find a project user
    ProjectUser newProjectUser = projectUsers.Find("AdminUser");
    if (newProjectUser == null)
    {
        // Create a project user
        newProjectUser = projectUsers.Create("AdminUser", GetSecureString("AdminUser123#"));
    }
    // Iterate through existing project users
    foreach (ProjectUser projectUser in projectUsers)
    {
        string projectName = projectUser.Name;
        // Indicates whether the project user is Active
        var isActive = projectUser.IsActive;
        //Change password of project user
        projectUser.SetPassword(GetSecureString("NewPassword@123"));
        //Change AuthenticationType of project user
        projectUser.ProjectUserAuthenticationType = AuthenticationType.Radius;
        // Deactivate project user
        projectUser.Deactivate();
        // Add a Custom Role to a deactivated project user
        projectUser.Roles.Add(customRole);
        // Add a System Role to a deactivated project user
        projectUser.Roles.Add(systemRole);
        // Remove a Custom Role from a deactivated project user
        bool successfullyRemoved = projectUser.Roles.Remove(customRole);
    }
}
```

5.16 Funktionen für den Zugriff auf Daten der Benutzerverwaltung

```
// Remove a System Role from a deactivated project user
isSuccessfullyRemoved = projectUser.Roles.Remove(systemRole);
// Activate project project user
projectUser.Activate();
// Delete the project user instance
projectUser.Delete();
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

5.16.2 Funktionen für globale UMAC-Benutzer und UMC-Server

5.16.2.1 Authentifizierung für die Verbindung mit einem UMC-Server

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit dem Dienst UmcServerConfigurator können Sie von der Eigenschaft UmcServer die Liste der UMC-Server abrufen. Das Authentifizierungsereignis wird auf dem UMC-Server aufgerufen, um die Verbindung zum Server herzustellen.

Der Dienst UmcServerConfigurator ist sowohl im Modus für geschützte Projekte als auch im Modus für nicht geschützte Projekte verfügbar. Das Authentifizierungsereignis wird nicht während eines Imports von UMC-Benutzern oder -Benutzergruppen vom UMC-Server ausgelöst, wenn der UMC-Admin als Benutzer angemeldet ist.

Das Ereignis wird nur ausgelöst, wenn der angemeldete Benutzer kein UMC-Benutzer mit einer Berechtigung "UMC View" auf dem UMC-Server ist. Dies kann zum Beispiel ein Projektbenutzer oder UMC-Benutzer ohne Berechtigung "UMC View" auf dem UMC-Server sein.

In TIA Portal Openness V17 ist nur ein UMC-Server verfügbar und vorkonfiguriert.

Programmcode

```
private static SecureString GetSecureString(string value)
{
    SecureString secureStr = new SecureString();
    if (!string.IsNullOrEmpty(value))
    {
        for (int i = 0; i < value.Length; i++)
        {
            secureStr.AppendChar(value[i]);
        }
    }
    return secureStr;
}
```

```
Project project = ...;
var umcServerConfigurator = project.GetService<UmcsServerConfigurator>();
UmcsServer umcServer = umcServerConfigurator.UmcsServer;
umcServer.Authentication += UmcsServer_Authentication;
```

UmcsServer_Authentication() besitzt EventArgs für die Übergabe des UMC-Administratorkennens und -Passworts.

```
private static void UmcsServer_Authentication(object sender, UmcsAuthenticationEventArgs e)
{
    e.UmcsCredentials.Name = "Admin";
    Console.WriteLine("Name is set");
    e.UmcsCredentials.SetPassword(GetSecureString("Admin123"));
    Console.WriteLine("Password is set");
}
```

5.16.2.2 Eine UMC-Benutzergruppe von einem UMC-Server abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness eine UMC-Benutzergruppe von einem UMC-Server abrufen. Die Methode GetUserGroupByName () wird auf dem UMC-Server verwendet, um die UMC-Benutzergruppe des angeschlossenen UMC-Servers abzurufen.

Wenn Methode `GetUserGroupByName ( )` angestoßen wird, wird durch den Ereignisrückruf die Authentifizierung am UMC-Server ausgelöst. Nur UMC-Benutzer mit der Berechtigung "UMC View" können eine UMC-Benutzergruppe vom UMC-Server abrufen.

Programmcode

```
Project project = ...;
var umcServerConfigurator = project.GetService<UmcServerConfigurator>();
UmcServer umcServer = umcServerConfigurator.UmcServer;
umcServer.Authentication += UmcServer_Authentication;
string umcUserGroupName = "xxxxxx";
Siemens.Engineering.Umac.UmcUserGroupInfo umcUserGroupsFromUmcServer =
umcServer.GetUserGroupByName(umcUserGroupName);
```

5.16.2.3 Einen UMC-Benutzer von einem UMC-Server abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness einen UMC-Benutzer von einem UMC-Server abrufen. `GetUserByName ( )` wird auf dem UMC-Server verwendet, um den entsprechenden UMC-Benutzer anhand des UMC-Benutzernamens vom UMC-Server abzurufen.

Wenn Methode `GetUserByName ( )` angestoßen wird, wird durch den Ereignisrückruf die Authentifizierung am UMC-Server ausgelöst. Nur ein UMC-Benutzer mit der Berechtigung "UMC View" kann UMC-Benutzer auf dem UMC-Server abfragen.

Programmcode

```
Project project = ...;
var umcServerConfigurator = project.GetService<UmcServerConfigurator>();
UmcServer umcServer = umcServerConfigurator.UmcServer;
umcServer.Authentication += UmcServer_Authentication;
string umcUserName = "xxxxxx";
Siemens.Engineering.Umac.UmcUserInfo umcUsersFromUmcServer =
umcServer.GetUserByName(umcUserName);
```

Hinweis

In TIA Portal Openness werden alle UMC-Ausnahmen als `Siemens.Engineering.TargetInvocationException` behandelt.

5.16.2.4 Eine UMC-Benutzergruppe hinzufügen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness eine UMC-Benutzergruppe in einem TIA Portal-Projekt auf einem UMC-Server hinzufügen.

Programmcode

```
Project project = ...;
var umcServerConfigurator = project.GetService<UmcServerConfigurator>();
var umacConfigurator = project.GetService<UmacConfigurator>();
UmcServer umcServer = umcServerConfigurator.UmcServer;
umcServer.Authentication += UmcServer_Authentication;
string umcUserGroupName = "xxxxxxx";
Siemens.Engineering.Umac.UmcUserGroupInfo umcUserGroupInfo =
umcServer.GetUserGroupByName(umcUserGroupName);
//Get the UmcUserGroupComposition.
UmcUserGroupComposition umcUserGroupComposition = umacConfigurator.UmcUserGroups;
//Create the selected UMC UserGroup in to the TIA Portal project
UmcUserGroup umcUserGroup = umcUserGroupComposition.Create(umcUserGroupInfo);
```

5.16.2.5 Alle UMC-Gruppen in TIA Portal abrufen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können mit TIA Portal Openness alle UMC-Benutzergruppen, die in TIA Portal-Projekten hinzugefügt wurden, abrufen.

Sind keine UMC-Benutzergruppen angelegt, gibt TIA Portal Openness eine leere Sammlung zurück.

Programmcode

```
//Get UmcUserGroups from the Project.  
UmcUserGroupComposition umcUserGroupComposition = umacConfigurator.UmcUserGroups;
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.16.2.6 Einen UMC-Benutzer in TIA Portal hinzufügen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness einen UMC-Benutzer einem TIA Portal-Projekt hinzufügen, nachdem die UMC-Benutzer vom UMC-Server abgerufen wurden.

Programmcode

```
Project project = ...;
var umcServerConfigurator = project.GetService<UmcServerConfigurator>();
var umacConfigurator = project.GetService<UmacConfigurator>();
UmcServer umcServer = umcServerConfigurator.UmcServer;
umcServer.Authentication += UmcServer_Authentication;
string umcUserName = "xxx";
Siemens.Engineering.Umac.UmcUserInfo umcUserInfo = umcServer.GetUserByName(umcUserName);
//Gets the UmcUserComposition
UmcUserComposition umcUserComposition = umacConfigurator.UmcUsers;
//Create the selected UMC user into the TIA Portal project
UmcUser umcUser = umcUserComposition.Create(umcUserInfo);
```

Siehe auch

Projekt öffnen (Seite 131)

5.16.2.7 Alle UMC-Benutzer in einem TIA Portal-Projekt abrufen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können mit TIA Portal Openness alle UMC-Benutzer in TIA Portal abrufen.

Sind keine UMC-Benutzer angelegt, gibt TIA Portal Openness eine leere Sammlung zurück.

Programmcode

```
var umacConfigurator = project.GetService<UmacConfigurator>();
UmcUserComposition umcUserComposition = umacConfigurator.UmcUsers;
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.16.2.8 Hinzugefügten UMC-Benutzer in TIA Portal suchen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können mit TIA Portal Openness einen UMC-Benutzer in einem TIA Portal-Projekt suchen.

Programmcode

```
string UmcUserName ="XXXX";  
UmcUserComposition umcUsersList = umacConfigurator.UmcUsers;  
UmcUser umcUser = umcUsersList.Find(UmcUserName);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.16.2.9 Hinzugefügte UMC-Benutzergruppe in einem TIA Portal-Projekt suchen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness eine UMC-Benutzergruppe in einem TIA Portal-Projekt suchen.

Programmcode

```
//Get UmcUserGroups from the Project.  
string UmcUserGroupName = "XXXX";  
UmcUserGroupComposition umcUserGroupList = umacConfigurator.UmcUserGroups;  
UmcUserGroup umcUserGroup = umcUserGroupList.Find(UmcUserGroupName);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.16.2.10 Einem UMC-Benutzer eine Rolle zuweisen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness einem im TIA Portal-Projekt hinzugefügten UMC-Benutzer eine Rolle zuweisen.

Programmcode

```
//Assign the System Role.  
UmacConfigurator umacConfigurationBuilder = ...;  
string s_SystemRoleName = "";  
SystemRole engineeringSystemRole =  
umacConfigurationBuilder.SystemRoles.Find(s_SystemRoleName);  
UmcUser umcUser = ...;  
umcUser.Roles.Add(engineeringSystemRole);  
//Assign the Custom Role.  
UmacConfigurator umacConfigurator = ...;  
CustomRole customRole = umacConfigurator.CustomRoles.Create("CustomRole");
```

Siehe auch

Projekt öffnen (Seite 131)

5.16.2.11 Liste zugewiesener Rollen für UMC-Benutzer und UMC-Benutzergruppen abrufen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness eine Liste der zugewiesenen Rollen für UMC-Benutzer und UMC-Benutzergruppen abrufen.

Programmcode

```
//Get assigned roles from UmcUserGroup.  
string UmcUserGroupName = "XXXX";  
UmcUserGroupComposition umcUserGroupList = umacConfigurationBuilder.UmcUserGroups;  
UmcUserGroup umcUserGroup = umcUserGroupList.Find(UmcUserGroupName);  
RoleAssociation assignedRolesInUmcUserGroup = umcUserGroup.Roles;  
//Get assigned roles from UmcUser.  
string UmcUserName = "XXXX";  
UmcUserComposition umcUserList = umacConfigurationBuilder.UmcUsers;  
UmcUser umcUser = umcUserList.Find(UmcUserGroupName);  
RoleAssociation assignedRolesInUmcUser = umcUser.Roles;
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

5.16.2.12 Einer UMC-Benutzergruppe eine Rolle zuweisen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness einer im TIA Portal-Projekt hinzugefügten UMC-Benutzergruppe eine Rolle zuweisen.

Programmcode

```
//Assign the System Role.
UmacConfigurator umacConfigurationBuilder = ...;
string s_SystemRoleName = "";
SystemRole engineeringSystemRole =
umacConfigurationBuilder.SystemRoles.Find(s_SystemRoleName);
UmcUserGroup umcUserGroup = ...;
umcUserGroup.Roles.Add(engineeringSystemRole);
//Assign the Custom Role
UmacConfigurator umacConfigurator = ...;
CustomRole customRole = umacConfigurator.CustomRoles.Create("CustomRole");
umcUserGroup.Roles.Add(customRole);
```

Siehe auch

Projekt öffnen (Seite 131)

5.16.2.13 Einem UMC-Benutzer und einer UMC-Benutzergruppe eine Rolle entziehen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal einem im TIA Portal-Projekt hinzugefügten UMC-Benutzer oder einer UMC-Benutzergruppe eine Rolle wieder entziehen.

Programmcode

Ändern Sie folgenden Programmcode, um einem im TIA Portal-Projekt hinzugefügten UMC-Benutzer eine Rolle zu entziehen.

```
//Remove the System Role.
bool isSuccessfullyRemoved = umcUser.Roles.Remove(engineeringSystemRole);
//Remove the Custom Role
isSuccessfullyRemoved = umcUser.Roles.Remove(customRole);
```

5.16 Funktionen für den Zugriff auf Daten der Benutzerverwaltung

Ändern Sie folgenden Programmcode, um einer im TIA Portal-Projekt hinzugefügten UMC-Benutzergruppe eine Rolle zu entziehen.

```
//Un assign the System Role.  
bool isSuccessfullyRemoved = umcUserGroup.Roles.Remove(engineeringSystemRole);  
//Un assign the Custom Role.  
isSuccessfullyRemoved = umcUserGroup.Roles.Remove(customRole);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.16.2.14 UMC-Benutzer und UMC-Benutzergruppe aktivieren/deaktivieren

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts

Verwendung

Sie können mit TIA Portal Openness einen UMC-Benutzer bzw. eine UMC-Benutzergruppe aktivieren / deaktivieren.

Programmcode

```
//Activate UMC User  
UmcUser umcUserToActivate = umacConfigurator.UmcUsers.Find("UserNameToActivate");  
umcUserToActivate.Activate();  
//Deactivate UMC User  
UmcUser umcUserToDeactivate = umacConfigurator.UmcUsers.Find("UserNameToDeactivate");  
umcUserToDeactivate.Deactivate();  
//Activate UMC User Group  
UmcUserGroup umcUserGroupToActivate =  
umacConfigurator.UmcUserGroups.Find("UserGroupNameToActivate");  
umcUserGroupToActivate.Activate();  
//Deactivate UMC User Group  
UmcUserGroup umcUserGroupToDeactivate =  
umacConfigurator.UmcUserGroups.Find("UserGroupNameToDeactivate");  
umcUserGroupToDeactivate.Deactivate();
```

5.16.2.15 Status eines UMC-Benutzers bzw. einer UMC-Benutzergruppe prüfen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness den Aktivierungsstatus eines UMC-Benutzers bzw. einer UMC-Benutzergruppe abrufen.

Programmcode

```
//Indicates whether the User is Active
UmcUser umcUserToActivate = umacConfigurator.UmcUsers.Find("UserNameToActivate");
var isUserActive = umcUserToActivate.IsActive;
//Indicates whether the Group is Active
UmcUserGroup umcUserGroupToDeactivate =
umacConfigurator.UmcUsersGroup.Find("UserGroupNameToActivate");
var isGroupActive = umcUserGroupToDeactivate.IsActive;
```

Siehe auch

Projekt öffnen (Seite 131)

5.16.2.16 UMC-Benutzer und UMC-Benutzergruppe löschen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness einen UMC-Benutzer und eine UMC-Benutzergruppe aus einem TIA Portal-Projekt löschen.

Programmcode

```
//Delete user from TIA Portal project.
UmUser umUserToDelete = umacConfigurator.UmUsers.Find("UserNameToDelete");
umUserToDelete.Delete();
//Delete of user group from TIA Portal project.
UmUserGroup umUserGroupToDelete =
umacConfigurator.UmUserGroups.Find("GroupNameToDelete");
umUserGroupToDelete.Delete();
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

5.16.2.17 UMC-Benutzer synchronisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können in TIA Portal Openness mit Synchronize() die UMC-Daten synchronisieren. Synchronize() ruft den UMC-Benutzer und die UMC-Benutzergruppe im Projekt ab, wenn der im Projekt vorhandene UMC-Benutzer und die UMC-Benutzergruppe im Domänenprojekt konfiguriert ist.

Programmcode

```
private void SynchronizeUMCUser()
{
m_UmcServerConfigurator = project.GetService<UmServerConfigurator>();
//Synchronize UMC Data
m_UmcServerConfigurator.Synchronize();
}
```


5.16.2.18 Konsistenz der UMC-Daten überprüfen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können in TIA Portal Openness mit `CheckConsistency()` die Konsistenz von UMC-Daten überprüfen. `CheckConsistency()` gibt "True" zurück, wenn die UMC-Daten im Projekt mit dem konfigurierten Server übereinstimmen.

Programmcode

```
private void CheckUMCData()
{
    m_UmcServerConfigurator = project.GetService<UmcServerConfigurator>();
    //Check consistency of UMC Data
    m_UmcServerConfigurator.CheckConsistency();
}
```

5.16.2.19 Runtime-Sitzungs-Timeout festlegen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können in TIA Portal Openness den Runtime-Timeout für einen Benutzer im UMAC Editor aktivieren oder deaktivieren.

Methoden und Parameter

Methodenname	Beschreibung
<code>ActivateRuntimeSessionTimeout()</code>	Runtime-Sitzungs-Timeout für einen Benutzer im UMAC Editor aktivieren
<code>DeactivateRuntimeSessionTimeout()</code>	Runtime-Sitzungs-Timeout für einen Benutzer im UMAC Editor deaktivieren

Name der Eigenschaft	Datentyp	Beschreibung
IsRuntimeSessionTimeoutActive	Boolean	Ruft den aktuellen Aktivierungsstatus der Eigenschaft Runtime-Sitzungs-Timeout ab. Ist er aktiv, gibt IsRuntimeSessionTimeoutActive "True" zurück. Andernfalls wird "False" zur Deaktivierung zurückgegeben.

Programmcode

```
private void SettingRuntimeSession()
{
    var umacConfigurator = Project.GetService<UmacConfigurator>();
    var projectUsers = umacConfigurator.ProjectUsers;
    var createdProjectUser = projectUsers.Create("ProjectUser1",
        "ProjectUser123".ToSecureString());
    //Gets the current activation state of runtime session timeout property, true if active and
    false if deactivated
    createdProjectUser.IsRuntimeSessionTimeoutActive;
    //deactivates the runtime session timeout property
    //cannot deactivate if already deactivated
    createdProjectUser.DeactivateRuntimeSessionTimeout();
    //Activates the runtime session timeout property
    //cannot activate if already activated
    createdProjectUser.ActivateRuntimeSessionTimeout();
}
```

5.16.2.20 Passwortrichtlinien für UMAC festlegen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können in TIA Portal Openness mit `passwordPolicyConfigurator()` Passwortrichtlinien für die Einstellungen der Runtime- und Engineering-Passwörter festlegen.

Eigenschaft

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
IncludesLowerCaseAndUpperCaseCharacters	Boolean	Legt den Wert für includesLowerCaseAndUpperCaseCharacters fest oder ruft ihn ab	Lesen/Schreiben
MinimumNumericCharacterLength	Short	Legt den Wert für minimumNumericCharacterLength im Passwort fest oder ruft ihn ab	Lesen/Schreiben
MinimumLength	Short	Legt den Wert für minimumLength des Passworts fest oder ruft ihn ab	Lesen/Schreiben
MinimumSpecialCharacterLength	Short	Legt den Wert für minimumSpecialCharacterLength im Passwort fest oder ruft ihn ab	Lesen/Schreiben
EnablePasswordAging	Short	Legt den Wert für enablePasswordAging fest oder ruft ihn ab	Lesen/Schreiben
MinimumUserPasswordsBlockedForReuse	Short	Legt den Wert für minimumUserPasswordBlockedForReuse fest oder ruft ihn ab	Lesen/Schreiben
PasswordValidityPrewarningTime	Short	Legt den Wert für passwordValidityPrewarningTime in Tagen fest, der außerhalb des Bereichs liegt	Lesen/Schreiben
PasswordValidity	Short	Legt den Wert für passwordValidity in Tagen fest, der außerhalb des Bereichs liegt	Lesen/Schreiben

Programmcode

```

private void PasswordPolicySetting()
{
    var passwordPolicyConfigurator = tiaProject.GetService<PasswordPolicyConfigurator>();
    //Sets the value of includesLowerCaseAndUpperCaseCharacters
    var includesLowerCaseAndUpperCaseCharacters = true;
    passwordPolicyConfigurator.IncludeLowerCaseAndUpperCaseCharacters =
    includesLowerCaseAndUpperCaseCharacters;
    //Gets the value of includesLowerCaseAndUpperCaseCharacters
    includesLowerCaseAndUpperCaseCharacters =
    passwordPolicyConfigurator.IncludeLowerCaseAndUpperCaseCharacters;
    // Sets the value of minimumNumericCharacterLength in password
    var minimumNumericCharacterLength = 2;
    passwordPolicyConfigurator.MinimumNumericCharacterLength = minimumNumericCharacterLength;
    // Gets the value of minimumNumericCharacterLength in password
    minimumNumericCharacterLength = passwordPolicyConfigurator.MinimumNumericCharacterLength;
    //Sets and gets the value of minLength of the password
    var minLength = 11;
    passwordPolicyConfigurator.MinimumLength = minLength;
    //Gets the value of minLength of the password
    minLength = passwordPolicyConfigurator.MinimumLength;
    //Sets the value of minimumSpecialCharacterLength in password
    var minimumSpecialCharacterLength = 1;
    passwordPolicyConfigurator.MinimumSpecialCharacterLength = minimumSpecialCharacterLength;
    //Gets the value of minimumSpecialCharacterLength in password
    minimumSpecialCharacterLength = passwordPolicyConfigurator.MinimumSpecialCharacterLength;
    //Sets the value of enablePasswordAging
    var enablePasswordAging = true;
    passwordPolicyConfigurator.EnablePasswordAging = enablePasswordAgingtrue;
    //Gets the value of enablePasswordAging
    enablePasswordAging = passwordPolicyConfigurator.EnablePasswordAging;
    //Sets the value of passwordValidity in days
    var passwordValidity = 5;
    passwordPolicyConfigurator.PasswordValidity = passwordValidity;
    //Gets the value of passwordValidity in days
    passwordValidity = passwordPolicyConfigurator.PasswordValidity;
    //Sets the value of passwordValidityPrewarningTime in days
    var passwordValidityPrewarningTime = 1;
    passwordPolicyConfigurator.PasswordValidityPrewarningTime =
    passwordValidityPrewarningTime;
    //Gets the value of passwordValidityPrewarningTime in days
    passwordValidityPrewarningTime =
    passwordPolicyConfigurator.PasswordValidityPrewarningTime;
    // Sets the value of minimumUserPasswordsBlockedForReuse
    var minimumUserPasswordsBlockedForReuse = 2;
    passwordPolicyConfigurator.MinimumUserPasswordsBlockedForReuse =
    minimumUserPasswordsBlockedForReuse;
    // Gets the value of minimumUserPasswordsBlockedForReuse
    minimumUserPasswordsBlockedForReuse =
    passwordPolicyConfigurator.MinimumUserPasswordsBlockedForReuse;
    //Sets the value of minLength of the password which is out of range
    var minLength = -11;
    try
    {
        passwordPolicyConfigurator.MinimumLength = minLength;
    }
}

```

5.16 Funktionen für den Zugriff auf Daten der Benutzerverwaltung

```
catch (Siemens.Engineering.PasswordPolicySettingsException exception)
{
    Console.WriteLine(exception.Message);
}
// Sets the value of minimumNumericCharacterLength in password which is out of range
var minimumNumericCharacterLength = -2;
try
{
    passwordPolicyConfigurator.MinimumNumericCharacterLength = minimumNumericCharacterLength;
}
catch (Siemens.Engineering.PasswordPolicySettingsException exception)
{
    Console.WriteLine(exception.Message);
}
//Sets the value of minimumSpecialCharacterLength in password which is out of range
var minimumSpecialCharacterLength = -1;
try
{
    passwordPolicyConfigurator.MinimumSpecialCharacterLength = minimumSpecialCharacterLength;
}
catch (Siemens.Engineering.PasswordPolicySettingsException exception)
{
    Console.WriteLine(exception.Message);
}
//Sets the value of passwordValidity in days which is out of range
var passwordValidity = -5;
try
{
    passwordPolicyConfigurator.PasswordValidity=passwordValidity;
}
catch (Siemens.Engineering.PasswordPolicySettingsException exception)
{
    Console.WriteLine(exception.Message);
}
//Sets the value of passwordValidityPrewarningTime in days which is out of range
var passwordValidityPrewarningTime = -1;
try
{
    passwordPolicyConfigurator.PasswordValidityPrewarningTime =
passwordValidityPrewarningTime;
}
catch (Siemens.Engineering.PasswordPolicySettingsException exception)
{
    Console.WriteLine(exception.Message);
}
// Sets the value of minimumUserPasswordsBlockedForReuse which is out of range
var minimumUserPasswordsBlockedForReuse = -2;
try
{
    passwordPolicyConfigurator.MinimumUserPasswordsBlockedForReuse=minimumUserPasswordsBloacke
dForReuse;
}
catch (Siemens.Engineering.PasswordPolicySettingsException exception)
{
    Console.WriteLine(exception.Message);
}
```

```
}
```

5.16.2.21 In geschütztem Projekt auf Anonymous User zugreifen

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Der Anonymous User benötigt für die Anmeldung im geschützten Projekt kein Passwort. Der Anonymous User wird standardmäßig im deaktivierten Zustand erstellt, wenn ein TIA Portal-Projekt geschützt ist. Bei einem geschützten Projekt wird nur ein Anonymous User erstellt.

Sie können mit TIA Portal Openness folgende Funktionen für den Anonymous User verwalten:

- Anonymous User aktivieren
- Anonymous User deaktivieren
- Anonymous User in geschütztem Projekt abrufen
- Anonymous User Systemrolle und benutzerdefinierte Rolle zuweisen bzw. vom AU entfernen

Programmcode

```
private void ManagingAnonymousUser()
{
    UmacConfigurator umacConfiguratorService = project.GetService<UmacConfigurator>();
    //Retrieve anonymous user. If not activated user will be nullvar anonymousUser =
    umacConfiguratorService.AnonymousUser;
    /activate/deactivate anonymous user
    umacConfiguratorService.ActivateAnonymousUser()
    //Retrieve anonymous user. If not activated user will be null
    anonymousUser = umacConfiguratorService.AnonymousUser;
    // Add a Custom or System Role to a Anonymous User
    anonymousUser.Roles.Add(customRoles.Find("CustomRole_01"));
    anonymousUser.Roles.Add(systemRoles.Find("SystemRole_01"));
    // Remove a Custom or System Role to a Anonymous User
    bool isSuccessfullyRemoved = anonymousUser.Roles.Remove(customRoles.Find("CustomRole_01"));
    isSuccessfullyRemoved = anonymousUser.Roles.Remove(systemRoles.Find("SystemRole_01"));
    umacConfiguratorService.DeactivateAnonymousUser();
}
```

5.16.2.22 Vollständiger Beispielcode

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Das folgende Skript für TIA Portal Openness beschreibt den kompletten Arbeitsablauf von Openness APIs für globale UMAC-Benutzer und UMC-Server.

5.16 Funktionen für den Zugriff auf Daten der Benutzerverwaltung

```

using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
using Siemens.Engineering.Umac;
namespace VollständigerBeispielcode
{
    internal class Program
    {
        static void Main(string[] args)
        {
            TiaPortal tiaPortal = new TiaPortal();
            //Registers for TIA Portal authentication for protected project opening
            tiaPortal.Authentication += TiaPortal_Authentication;
            Project umacProject = tiaPortal.Projects.Open(new FileInfo("ProtectedProject_Path"));
            var umcServerConfigurator = umacProject.GetService<UmcServerConfigurator>();
            // Getting UMC servers.
            UmcServer umcServer = umcServerConfigurator.UmcServer;
            // Register for UMC Server Authentication
            umcServer.Authentication += UmcServer_Authentication;
            var umacConfigurator = umacProject.GetService<UmacConfigurator>();
            // Use case :Import User Group, Add User Group to Tia portal project , Assign & Unassign
            Role to UserGroup
            // Delete the User group from Tia portal project.
            // Retrieve an UMC User Group from UMC Server
            const string UmcUserGroupName = "xxxxxx";
            UmcUserGroupInfo umcUserGroupInfo = umcServer.GetUserGroupByName(UmcUserGroupName);
            // Add an UMC User Group into TIA portal Project.
            UmcUserGroupComposition umcUserGroupComposition = umacConfigurator.UmcUserGroups;
            UmcUserGroup umcUserGroup = umcUserGroupComposition.Create(umcUserGroupInfo);
            // Assign and Unassign roles to User Group.
            //Assign the System Role.
            const string SystemRoleName = "Engineering administrator";
            SystemRole engineeringAdministratorSystemRole =
            umacConfigurator.SystemRoles.Find(SystemRoleName);
            umcUserGroup.Roles.Add(engineeringAdministratorSystemRole);

```

```

//Assign the Custom Role.
CustomRole customRole = umacConfigurator.CustomRoles.Create("CustomRole");
umcUserGroup.Roles.Add(customRole);
//Un assign the System Role.
umcUserGroup.Roles.Remove(engineeringAdministratorSystemRole);
//Un assign the Custom Role.
umcUserGroup.Roles.Remove(customRole);
//Delete the UMC user group in Tia Portal project.
UmcUserGroup umcUserGroupToDelete = umacConfigurator.UmcUserGroups.Find(UmcUserGroupName);
umcUserGroupToDelete.Delete();
// Use case :Import UMC User , Add UMC User to Tia portal project , Assign & Unassign Roles
to UMC User,
// Delete the UMC User from Tia portal project.
// Import UMC User from UMC Server
const string UmcUserName = "xxxxx";
UmcUserInfo umcUserInfo = umcServer.GetUserByName(UmcUserName);
//Add UMC User to Tia Portal project
UmcUserComposition umcUserComposition = umacConfigurator.UmcUsers;
UmcUser umcUser = umcUserComposition.Create(umcUserInfo);
// Assign & Unassign Roles to UMC User
//Assign the System Role.
const string SystemRoleNameForUser = "NET Radius";
SystemRole systemRoleForUmcUser = umacConfigurator.SystemRoles.Find(SystemRoleNameForUser);
umcUser.AssignRole(systemRoleForUmcUser); //Needs to be:
umcUser.Roles.Add(systemRoleForUmcUser);
//Assign the Custom Role.
CustomRole customRoleNameForUmcUser =
umacConfigurator.CustomRoles.Create("CustomRoleName");
umcUser.Roles.Add(customRoleNameForUmcUser);
//Un assign the System Role.
umcUser.Roles.Remove(systemRoleForUmcUser);
//Un assign the Custom Role.
umcUser.Roles.Remove(customRoleNameForUmcUser);
//Delete the UMC User from Tia Portal project
UmcUser umcUserToDelete = umacConfigurator.UmcUsers.Find(UmcUserName);
umcUserToDelete.Delete();
}
private static void TiaPortal_Authentication(object sender, AuthenticationEventArgs e)
{
e.AuthenticationTypeProvider = AuthenticationTypeProvider.Credentials;
e.UmacCredentials.Type = UmacUserType.Project;
e.UmacCredentials.Name = "Admin";
e.UmacCredentials.SetPassword(GetSecureString("Admin123"));
}
private static void UmcServer_Authentication(object sender, UmcAuthenticationEventArgs e)
{
e.UmcCredentials.Name = "Admin";
e.UmcCredentials.SetPassword(GetSecureString("Admin123"));
}
private static SecureString GetSecureString(string password)
{
SecureString secureString = new SecureString();
foreach (char c in password)
{
secureString.AppendChar(c);
}
}

```

```
return secureString;  
}  
}  
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.17 Funktionen auf OPC

5.17.1 Konfiguration des sicheren Kommunikationsprotokolls für den OPC UA-Server

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit der TIA Portal Openness-Anwendung den OPC UA-Server mit der Sicherheitsrichtlinie Basic256Sha256 konfigurieren. Die Sicherheitsrichtlinie Basic256Sha256 muss in den Runtime-Einstellungen hinzugefügt werden. RDP muss die Eigenschaften in der xml-Konfigurationsdatei übersetzen.

Die Standardwerte lauten Enabled, Sign sowie Sign and Encrypt.

In der XML-Datei <Project>\OPC\uaserver\OPCUaServerWinCCPro.xml müssen Sie die Sicherheitsrichtlinie sowie Sicherheitsrichtlinien gemäß der ES-Gerätekonfiguration festlegen.

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <KOPCUA_Server_WinCC xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ua="http://opcfounda
3
4  <SecuredApplication>
5    <BaseAddresses>
6      <ua:String>opc.tcp://[HostName]:4861</ua:String>
7    </BaseAddresses>
8    <SecurityProfileUris>
9      <SecurityProfile>
10        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#None</ProfileUri>
11        <Enabled>true</Enabled>
12      </SecurityProfile>
13      <SecurityProfile>
14        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic128Rsa15</ProfileUri>
15        <Enabled>true</Enabled>
16      </SecurityProfile>
17      <SecurityProfile>
18        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256</ProfileUri>
19        <Enabled>true</Enabled>
20      </SecurityProfile>
21      <SecurityProfile>
22        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256Sha256</ProfileUri>
23        <Enabled>true</Enabled>
24      </SecurityProfile>
25    </SecurityProfileUris>
26  </SecuredApplication>
27
28  <ServerConfiguration>
29    <SecurityPolicies>
30      <SecurityPolicy>
31        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#None</ProfileUri>
32        <MessageSecurityModes>None</MessageSecurityModes>
33      </SecurityPolicy>
34      <SecurityPolicy>
35        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic128Rsa15</ProfileUri>
36        <MessageSecurityModes>Sign</MessageSecurityModes>
37      </SecurityPolicy>
38      <SecurityPolicy>
39        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic128Rsa15</ProfileUri>
40        <MessageSecurityModes>SignAndEncrypt</MessageSecurityModes>
41      </SecurityPolicy>
42      <SecurityPolicy>
43        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256</ProfileUri>
44        <MessageSecurityModes>Sign</MessageSecurityModes>
45      </SecurityPolicy>
46      <SecurityPolicy>
47        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256</ProfileUri>
48        <MessageSecurityModes>SignAndEncrypt</MessageSecurityModes>
49      </SecurityPolicy>
50      <SecurityPolicy>
51        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256Sha256</ProfileUri>
52        <MessageSecurityModes>Sign</MessageSecurityModes>
53      </SecurityPolicy>
54      <SecurityPolicy>
55        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256Sha256</ProfileUri>
56        <MessageSecurityModes>SignAndEncrypt</MessageSecurityModes>
57      </SecurityPolicy>
58    </SecurityPolicies>

```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.17.2 Sicherheitsrichtlinie für OPC UA festlegen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Öffnen eines Projekts (Seite 131)
- OPC UA Server ist aktiviert

Einleitung

Sie können mit der TIA Portal Openness-Anwendung die Sicherheitsrichtlinie in OPC UA festlegen. Sie können die Sicherheitsrichtlinie als dynamisches Attribut vom Typ markierte Enumeration implementieren: `OpcUaSecurityPolicies`. Die Sicherheitsrichtlinie ist in TIA Portal Openness nur verfügbar, wenn der OPC UA Server aktiviert ist.

Wenn der OPC UA Server deaktiviert ist, wird bei dem Versuch, wegen eines anderen, nicht verfügbaren Attributs auf die Sicherheitsrichtlinie zuzugreifen, eine `EngineeringNotSupportedException` ausgelöst.

Die nachstehende Tabelle zeigt die möglichen Werte, die für Sicherheitsrichtlinien zu finden sind:

Name im TIA UI	Enumerations-Eintrag	Wert	Anmerkungen
-	NoneSelected	0	Entspricht der TIA UI, wenn kein Kontrollkästchen markiert ist.
No security	OpcUaSecurityPoliciesNone	1	
Basic128Rsa15 - Sign	OpcUaSecurityPolicies128RSAS	2	
Basic128Rsa15 - Sign & Encrypt	OpcUaSecurityPolicies128RSASE	4	
Basic256 - Sign	OpcUaSecurityPolicies256S	8	
Basic256 - Sign & Encrypt	OpcUaSecurityPolicies256SE	16	
Basic256Sha256 - Sign	OpcUaSecurityPolicies256SHAS	32	
Basic256Sha256 - Sign & Encrypt	OpcUaSecurityPolicies256SHASE	64	
Aes128 - Sha256 - RsaOaep - Sign	OpcUaSecurityPolicies256RSAOAEPS	128	
Aes128Sha256 - RsaOaep - Sign & Encrypt	OpcUaSecurityPolicies256RSAOAEPSE	256	
Aes256Sha256RsaPss - Sign	OpcUaSecurityPolicies256RSAPSSS	512	
Aes256Sha256RsaPss - Sign & Encrypt	OpcUaSecurityPolicies256RSAPSSSE	1024	

Programmcode

Um die Sicherheitsrichtlinie in OPC UA festzulegen, ändern Sie folgenden Programmcode:

```
DeviceItem UpcUaSubmodule = ...;
object SecurityPolicies = UpcUaSubmodule.GetAttribute("OpcUaSecurityPolicies");
if (SecurityPolicies | OpcUaSecurityPolicies.OpcUaSecurityPolicies256S ==
OpcUaSecurityPolicies.OpcUaSecurityPolicies256S)
{
    //Do something
}
UpcUaSubmodule.SetAttribute("OpcUaSecurityPolicies",
OpcUaSecurityPolicies.OpcUaSecurityPolicies256S |
OpcUaSecurityPolicies.OpcUaSecurityPolicies256SHASE);
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.17.3 Zugriff auf den OPC UA-Referenz-Namespace

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness auf den Referenz-Namespaces und den Subtyp der Serverschnittstelle (SIMATIC-Schnittstelle) zugreifen. Zu Informationen über die OPC UA SIMATIC-Schnittstelle siehe Zugriff auf die OPC UA Server-(SIMATIC)Schnittstelle (Seite 984)

Die TIA Portal Openness-Schnittstelle unterstützt die folgenden Funktionen des Referenz-Namespaces:

- Erstellen eines neuen Referenz-Namespaces
- Importieren eines vorhandenen Referenz-Namespaces
- Exportieren eines vorhandenen Referenz-Namespaces
- Löschen eines vorhandenen Referenz-Namespaces

Methode und Eigenschaft

Der Referenz-Namespaces unterstützt die folgenden Methoden in TIA Portal Openness:

Methode	Rückgabotyp	Definition
Create()	Siemens.Engineering.SW.OpcUa.ReferenceNamespace	Zum Erstellen eines neuen Referenz-Namespaces-Objekts, das die mit der Pfaddatei bereitgestellte XML-Datei enthält
Import()	System.Void	Zum Festlegen des Inhalts des Serverschnittstellenobjekts auf Grundlage der XML-Datei. Mit der Methode import() wird die XML-Datei anhand des unterstützten Schemas geparkt. Wenn die Datei erfolgreich geparkt werden kann, wird die Information dem Referenz-Namespaces hinzugefügt und die Importmethode ist erfolgreich. Wenn die Datei nicht erfolgreich geparkt werden kann, bleibt der Inhalt des Objekts unverändert und die Importmethode löst eine Ausnahme aus.
Export()	System.Void	Zum Schreiben des Inhalts des Referenz-Namespaces-Objekts in eine XML-Datei. Wenn der Referenz-Namespaces keine Daten enthält, wird eine Ausnahme ausgelöst.
Delete()	System.Void	Entfernt das Objekt aus der ReferenceNamespacesComposition, in der es enthalten ist, und hebt die Zuordnung des Objekts auf.

Create() in der Referenz-Namespaces-Klasse unterstützt die folgende Parameterdefinition:

Parameter	Rückgabotyp	Definition
name	System.String	Name für das neue Referenz-Namespaces-Objekt
xmlFilePath	System.IO.FileInfo	Pfad zur XML-Datei für den Import

Import() in der Referenz-Namespaces-Klasse unterstützt die folgende Parameterdefinition:

Parameter	Rückgabotyp	Definition
filePath	System.IO.FileInfo	Wrapper-Objekt für den Dateipfad der zu importierenden Datei

Export() in der Referenz-Namespaces-Klasse unterstützt die folgende Parameterdefinition:

Parameter	Rückgabotyp	Definition
filePath	System.IO.FileInfo	Wrapper-Objekt für den Dateipfad, in den die zu exportierende Datei geschrieben werden soll.

Der Referenz-Namespaces stellt die folgenden Eigenschaften zur Verfügung:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Author	System.String	Autor	Lesen/Schreiben
Comment	Siemens.Engineering.Multilingual.Text	Kommentar	Lesen/Schreiben
Creation-Time	System.DateTime	Erstellungszeitpunkt des Referenz-Namespaces	Schreibgeschützt

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Enabled	System.Boolean	Referenz-Namespace freigeben und in die PLC laden	Lesen/Schreiben
LastModified	System.DateTime	Zeitpunkt der letzten Änderung des Referenz-Namespaces	Schreibgeschützt
Name	System.String	Name	Schreibgeschützt

Programmcode

```

// Accessing OPC UA interface;
PlcSoftware software = ....;
OpcUaProvider provider = software.GetService<OpcUaProvider>();
if (provider != null)
{
    OpcUaCommunicationGroup commGroup = provider.CommunicationGroup;
    ServerInterfaceGroup serverInterfaceGroup = commFolder.ServerInterfaceGroup;
    ReferenceNamespaceComposition referenceNamespaces =
    serverInterfaceGroup.ReferenceNamespaces;
    // Iterating through ReferenceNamespace class

    foreach (var referenceNamespace in referenceNamespaces)
    {
        // ....;
    }
}
// Accessing ReferenceNamespace properties
private void accessReferenceNamespaceProperty()
{
    ReferenceNamespaceComposition referenceNameSpace = ...;
    if (referenceNamespaces != null)
    {
        // Creating reference namespace object containing the xml file provided with name
        ReferenceNamespace referenceNamespace = referenceNamespaces.Create(name, xmlFilePath);
        referenceNamespace.Author = "XYZ";
    }
}
// Import the reference namespace object
private void importReferenceNamespace()
{
    ReferenceNamespaceComposition referenceNamespaces = ...;
    if (referenceNamespaces != null)
    {
        ReferenceNamespace referenceNamespace = referenceNamespaces.Create(name, xmlFilePath);
        if (referenceNamespace != null)
        {
            referenceNamespace.Import(new FileInfo(importFilePath));
        }
    }
}
// Export the reference namespace object to XML file
private void exportReferenceNamespace()
{
    String exportFilePath = Path.Combine(Directory.GetCurrentDirectory(),
    "ExportInterface.xml")
    ReferenceNamespace referenceNamespace = ...;
    referenceNamespace.Export(new FileInfo(exportFilePath));
}
// Remove the reference namespace object
private void deleteReferenceNamespace()
{
    ReferenceNamespace referenceNamespace = ...;
    referenceNamespace.Delete();
}

```

Siehe auch

Projekt öffnen (Seite 131)

5.17.4 Zugriff auf die OPC UA Server-(SIMATIC)Schnittstelle

Voraussetzung

- Die Anwendung ist mit TIA Portal über TIA Portal Openness verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness auf den Subtyp der Serverschnittstelle (SIMATIC-Schnittstelle) zugreifen.

Die TIA Portal Openness-Schnittstelle unterstützt die folgenden Funktionen der SIMATIC-Schnittstelle:

- Erstellen einer SIMATIC-Schnittstelle
- Exportieren einer vorhandenen SIMATIC-Schnittstelle
- Löschen einer vorhandenen SIMATIC-Schnittstelle

Hinweis

Die S7-1200 FW V4.4 umfasst jetzt die Unterstützung für SIMATIC OPC UA-Schnittstellen.

Methode und Eigenschaft

Die Klasse der SIMATIC-Schnittstelle unterstützt die folgende Methode in TIA Portal Openness:

Methode	Rückgabetyt	Definition
Create()	Siemens.Engineering.SW.OpcUa.SimaticInterface	Zum Erstellen einer SIMATIC OPC UA-Schnittstelle zur PLC
Export()	System.Void	Zum Schreiben des Inhalts des SIMATIC-Schnittstellenobjekts in eine XML-Datei. Wenn die SIMATIC-Schnittstelle keine Daten enthält, wird eine Ausnahme ausgelöst.
Delete()	System.void	Entfernt das Objekt aus der Zusammenstellung der SIMATIC-Schnittstelle, in der es enthalten ist, und hebt die Zuordnung des Objekts auf

Create() in der SIMATIC-Schnittstelle unterstützt die folgende Parameterdefinition:

Parameter	Rückgabotyp	Definition
name	System.String	Name für das neue SIMATIC-Schnittstellenobjekt

Export() in der SIMATIC-Schnittstelle unterstützt die folgende Parameterdefinition:

Parameter	Rückgabotyp	Definition
filePath	System.IO.FileInfo	Wrapper-Objekt für den Dateipfad, in den die zu exportierende Datei geschrieben werden soll.

Die Klasse der SIMATIC-Schnittstelle unterstützt die folgenden Eigenschaften in TIA Portal Openness:

Name der Eigenschaft	Datentyp	Beschreibung	Zugriff
Author	System.string	Autor	Lesen/Schreiben
Comment	Siemens.Engineering.Multilingual.Text	Kommentar	Lesen/Schreiben
Creation-Time	System.DateTime	Erstellungszeitpunkt der SIMATIC-Schnittstelle	Schreibgeschützt
Enabled	System.Boolean	SIMATIC-Schnittstelle freigeben und in die PLC laden	Lesen/Schreiben
LastModified	System.DateTime	Zeitpunkt der letzten Änderung der SIMATIC-Schnittstelle	Schreibgeschützt
Name	System.String	Name	Schreibgeschützt

Programmcode

```
// Accessing OPC UA interface;
PlcSoftware software = ....;
OpcUaProvider provider = software.GetService<OpcUaProvider>();
if (provider != null)
{
    OpcUaCommunicationGroup commGroup = provider.CommunicationGroup;
    ServerInterfaceGroup serverInterfaceGroup = commFolder.ServerInterfaceGroup;
    SimaticInterfaceComposition simaticInterfaces = serverInterfaceGroup.SimaticInterfaces;
    // Iterating through Simatic Interfaces class

    foreach (var simaticInterface in simaticInterfaces)
    {
        // ....;
    }
}
// Accessing Simatic Interfaces properties
private void accessSimaticInterfaceProperty()
{
    SimaticInterfaceComposition simaticInterfaces = ...;
    if (simaticInterfaces != null)
    {
        // create a new empty SIMATIC interface object.
        SimaticInterface simaticInterface = simaticInterfaces.Create(name);
        simaticInterface.Author = "XYZ";
    }
}
// Export the Simatic Interface object to XML file
private void exportReferenceNamespace()
{
    String exportFilePath = Path.Combine(Directory.GetCurrentDirectory(),
    "ExportInterface.xml")
    SimaticInterface simaticInterface = ...;
    simaticInterface.Export(new FileInfo(exportFilePath));
}
// Remove the Simatic Interface object
private void deleteSimaticInterface()
{
    SimaticInterface simaticInterface = ...;
    simaticInterface.Delete();
}
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

5.17.5 Zugriff auf die OPC UA Server-Schnittstelle

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können Serverschnittstellenobjekte erstellen und XML-Dateien importieren, die die OPC UA-Serverschnittstelle für den PLC definieren.

Mit der TIA Portal Openness API können Sie über den OpcUaProvider -Dienst auf die folgenden Funktionalitäten zugreifen:

- Hinzufügen eines OPC UA-Serverschnittstellenobjekts
- Enumerieren über die Serverschnittstelle des PLC
- Importieren einer Serverschnittstellendatei
- Exportieren einer vorhandenen Serverschnittstelle nach XML
- Aktivieren/Deaktivieren der Serverschnittstelle
- Löschen einer Serverschnittstelle

Programmcode

Sie können über einen OpcProvider-Dienst, den Sie über PlcSoftware erwerben können, mit OPC UA-Serverschnittstellen interagieren.

```
PlcSoftware software = ....;
OpcUaProvider provider = software.GetService<OpcUaProvider>();
if (provider != null)
{
    ....
}
```

Um die Navigation zu den OPC UA-Serverschnittstellen zu ermöglichen, ändern Sie folgenden Programmcode:

```
PlcSoftware software = ....;
OpcUaProvider provider = software.GetService<OpcUaProvider>();
if (provider != null)
{
    OpcUaCommunicationGroup commGroup = provider.CommunicationGroup;
    ServerInterfaceGroup serverInterfaceGroup = commGroup.ServerInterfaceGroup;
    ServerInterfaceComposition serverInterfaces = serverInterfaceGroup.ServerInterfaces;
}
```

Um die Elemente in der Sammlung `ServiceInterfaceComposition` zu enumerieren, ändern Sie folgenden Programmcode:

```
ServerInterfaceComposition serverInterfaces = ....;
foreach (ServerInterface serverInterface in serverInterfaces)
{
    ....;
}
```

Um ein OPC UA-Serverschnittstellenobjekt hinzuzufügen, ändern Sie folgenden Programmcode:

```
ServerInterfaceComposition serverInterfaces = ....;
if (serverInterfaces != null)
{
    ServerInterface serverInterface = serverInterfaces.Create(name);
}
```

Um die Eigenschaften eines OPC UA-Serverschnittstellenobjekts abzurufen und festzulegen, ändern Sie folgenden Programmcode:

```
ServerInterfaceComposition serverInterfaces = ....;
if (serverInterfaces != null)
{
    ServerInterface serverInterface = serverInterfaces.Create("New Interface");
    serverInterface.Author = "James Cornett";
}
```

Um den Inhalt des Serverschnittstellenobjekts basierend auf einer XML-Datei festzulegen, ändern Sie folgenden Programmcode:

```
ServerInterfaceComposition serverInterfaces = ....;
if (serverInterfaces != null)
{
    ServerInterface serverInterface = serverInterfaces.Create(name);
    if (serverInterface != null)
    {
        serverInterface.Import(new FileInfo(importFilePath));
    }
}
```

Um den Inhalt des Serverschnittstellenobjekts in eine XML-Datei zu schreiben, ändern Sie folgenden Programmcode:

```
String exportFilePath = Path.Combine(Directory.GetCurrentDirectory(),
"ExportInterface.xml");
ServerInterface serverInterface = ...;
serverInterface.Export(new FileInfo(exportFilePath));
```


Um das Objekt aus der `ServerInterfaceComposition` zu entfernen, in der es enthalten ist, und die Zuordnung des Objekts aufzuheben, ändern Sie folgenden Programmcode:

```
ServerInterface serverInterface = ...;
serverInterface.Delete();
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

5.18 SiVArc Openness

5.18.1 Einleitung

Einleitung

Mit der TIA Portal Openness-Anwendung können Sie SiVArc instanziiieren. Sie benötigen eine Client-Anwendung für den Zugriff auf das TIA Portal und starten dann über die Openness-Funktion die SiVArc-Dienste. Weitere Einzelheiten zur Einrichtung und zum Zugriff auf Openness finden Sie im Benutzerhandbuch zum TIA Portal.

Einrichten der Anwendung

Um eine Client-Anwendung einzurichten, gehen Sie wie folgt vor:

1. Erstellen Sie eine Konsolenanwendung. Fügen Sie einen Verweis auf die Public API (`Siemens.Engineering.dll`) hinzu, verfügbar am Speicherort der installierten Binärdateien unter `PublicAPI\19\Siemens.Engineering.dll`.
2. Fügen Sie der Konfigurationsdatei `Konfigurationsdetails` hinzu. Ausführliche Informationen über die Konfigurationsdetails und den Zugriff auf die Public API finden Sie im TIA Openness Wiki.

3. Für den Zugriff auf den SiVArc-Dienst verwenden Sie die nachfolgend erwähnte API:

```
using (TiaPortal tia = new
TiaPortal(TiaPortaMode.WithUserInterface))
{
    Project myProject = tia.Projects.Open(new
FileInfo(@"D:\Project\Project.ap19"));
    SivarC sivarC = myproject?.GetService<SivarC>();
    if (sivarC != null)
    {
    }
}
```

5.18.2 Openness Lizenz-Handhabung

Wenn ein Benutzer ein TIA-Projekt über Openness API öffnet und über Openness API SiVArc-Datenänderungen vornimmt (Regeln und Generierung), validiert das System die Lizenz und wenn die Lizenz nicht gefunden wird, wird die wiederherstellbare Ausnahme `LicenseNotFound` zurückgegeben.

5.18.3 SiVArc-Instanzeigenschaften

SiVArc-Instanzeigenschaften

Die nachfolgende Tabelle führt die unterstützten Eigenschaften und Methoden für SiVArc auf:

Name der Eigenschaft	Beschreibung	Datentyp
AlarmRules	Ankerobjekt für alle Alarmregelobjekte	AlarmRulesBrowsable
ScreenRules	Ankerobjekt für alle Bildregelobjekte	ScreenRulesBrowsable
TextlistRules	Ankerobjekt für alle Textlistenregelobjekte	TextlistRulesBrowsable
TagRules	Ankerobjekt für alle Variablenregelobjekte	TagRulesBrowsable
CopyRules	Ankerobjekt für alle Kopierregelobjekte	CopyRulesBrowsable
TextlistRules	Enumeration sämtlicher direkter Textlistenregeln der ersten Ebene	TextlistRuleComposition
Alarm Rules	Enumeration sämtlicher direkter Alarmregeln der ersten Ebene	AlarmRuleComposition
ScreenRules	Enumeration sämtlicher direkter Bildregeln der ersten Ebene	ScreenRuleComposition
TagRules	Enumeration sämtlicher direkter Variablenregeln der ersten Ebene	TagRuleComposition
CopyRules	Enumeration sämtlicher direkter Kopierregeln der ersten Ebene	CopyRuleComposition
Groups	Enumeration sämtlicher direkter Alarmregelgruppen der ersten Ebene	AlarmRuleGroupComposition
ScreenRules-Groups	Enumeration sämtlicher direkter Bildregelgruppen der ersten Ebene	ScreenRuleGroupComposition
TextlistGroups	Enumeration sämtlicher direkter Textlistenregelgruppen der ersten Ebene	TextlistRuleGroupComposition
TagRulesGroups	Enumeration sämtlicher direkter Variablenregelgruppen der ersten Ebene	TagRuleGroupComposition
CopyRulesGroups	Enumeration sämtlicher direkter Kopierregelgruppen der ersten Ebene	CopyRuleGroupComposition

In der folgenden Tabelle wird die Zusammensetzung von `AlarmRule` und `AlarmRuleGroup` aufgeführt. Gleiches gilt für andere SiVArc-Objekte.

Methodenname	Parameter	Beschreibung	Datentyp
Find	Zeichenkette – Alarmregel-/Regelgruppenname	Sucht die Alarmregel/ Alarmregelgruppe in der Sammlung der Alarmregeln/Alarmgruppen	AlarmRule
CreateFrom	MasterCopy – Kopiervorlage Alarmregel/Alarmregelgruppe	Kopiert die Kopiervorlage der Alarmregel/ Alarmregelgruppe aus der Bibliothek in das Projekt und verwendet die Standardoption zum Ersetzen	AlarmRule
CreateFrom	MasterCopy – Kopiervorlage Alarmregel/Alarmregelgruppe, CreateOptions – Umbenennen/ Ersetzen	Kopiert die Kopiervorlage der Alarmregel/ Alarmregelgruppe aus der Bibliothek in das Projekt und verwendet die Option zum Erstellen	AlarmRule

Openness unterstützt die Generierung und Konfiguration von Alarmen über Alarmregeln für Unified-Geräte.

5.18.4 Kopieren von Regeln oder Gruppen aus der Bibliothek

Voraussetzung

- Starten Sie die TIA Portal Openness-Anwendung. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- Ein TIA Portal-Projekt mit Bildregeln, Bildregelgruppen, Kopiervorlage der Bildregel und Kopiervorlage der Bildregelgruppe ist vorhanden.

Fall 1: Wenn Sie Regeln/Regelgruppen aus der Kopiervorlage in die Bildregeltabelle kopieren

Mit der API "CreateFrom" können Sie die Kopiervorlage der Regeln/Regelgruppen von der Bibliothek in den Regeleditor kopieren. Wenn erfolgreich, gibt die API-Funktion die ScreenRule/ScreenRuleGroup zurück. Der folgende Code zeigt, wie Regeln oder Regelgruppen aus der Kopiervorlage in die SiVArc-Regeltabelle kopiert werden:

// Regelkopiervorlage aus der Bibliothek in die Regeltabelle kopieren

```
MasterCopy screenRuleMasterCopy =
project.ProjectLibrary.MasterCopyFolder.MasterCopies.Find("Screen rule_1");
if (screenRuleMasterCopy != null)
{
var rule = sivarC.ScreenRules.Tables.Find("Default screen rule
table").Rules.CreateFrom(screenRuleMasterCopy);
if (rule != null)
{
Console.WriteLine("Copied screen rule name - " + rule.Name);
Console.WriteLine("Copied screen rule comment - " + rule.Comment);
Console.WriteLine("Copied screen rule enabled - " + rule.Enabled);
Console.WriteLine("Copied screen rule condition - " + rule.Condition);
}
}
```

Standardmäßig ist das Verhalten "Ersetzen".

Fall 2: Wenn Sie Regeln/Regelgruppen aus der Kopiervorlage kopieren, können die vorhandenen Regeln/Regelgruppen basierend auf der zweiten Parametervorgabe umbenannt oder ersetzt werden.

Wenn Regeln/Regelgruppen der Kopiervorlage bereits in der SiVArc-Regeltabelle vorhanden sind und Sie versuchen, diese zu kopieren, werden die Regeln/Regelgruppen umbenannt. Die API "CreateOptions" erstellt die Regeln/Regelgruppen im SiVArc-Editor, sofern sie nicht vorhanden sind, und ersetzt andernfalls bereits vorhandene Regeln/Regelgruppen. Wenn erfolgreich, benennt die API-Funktion die ScreenRule/ScreenRuleGroup um. Das folgende Code-Snippet zeigt das Ersetzen von Regeln/Regelgruppen:

```
MasterCopy screenRuleMasterCopy =
project.ProjectLibrary.MasterCopyFolder.MasterCopies.Find("Screen rule_1");
if (screenRuleMasterCopy != null)
{
var rule = sivarC.ScreenRules.Tables.Find("Default screen rule
table").Rules.CreateFrom(screenRuleMasterCopy, CreateOptions.Rename);
if (rule != null)
{
Console.WriteLine("Copied screen rule name - " + rule.Name);
Console.WriteLine("Copied screen rule comment - " + rule.Comment);
Console.WriteLine("Copied screen rule enabled - " + rule.Enabled);
Console.WriteLine("Copied screen rule condition - " + rule.Condition);
}
}
```

Beim Erstellen von Kopierregeln können Sie die Ordner mit Kopiervorlagen in den Ordnern Projektbibliothek und Globale Bibliothek durchsuchen. Zum Kopieren von Regeln rufen Sie SiVArc mit GetService auf. Die API CopyRule kopiert das Objekt mit Hilfe der API

MasterCopyFolder und ProjectLibrary in den Ordner mit den Kopiervorlagen. Das folgende Code-Snippet zeigt, wie Sie Ordner mit der Kopierregel durchsuchen können:

```
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
string projectPath = @"D:\Temp\Project\Project.ap19";
Project project = portal.Projects.Open(new
System.IO.FileInfo(projectPath));
Sivarc sivarc = project.GetService<Sivarc>();
if (sivarc != null)
{
CopyRule copyRule = sivarc.CopyRules.Tables.Find("Default copy rule
table").Rules.Find("Copy rule_1");
MasterCopyFolder masterCopyFolder =
project.ProjectLibrary.MasterCopyFolder.Folders.Find("CopyRuleFolder");
if (masterCopyFolder != null && copyRule != null
{
copyRule.LibraryObject = masterCopyFolder;
}
}
```

Bei Kopierregeln können Sie die Ordner mit Kopiervorlagen in den Ordnern Projektbibliothek und Globale Bibliothek durchsuchen. Zum Kopieren von Regeln rufen Sie SiVArc mit GetService auf. Die API CopyRule kopiert das Objekt in den Ordner Kopiervorlagen mit Hilfe des copyrule.LibraryObject. Das System zeigt eine Ausnahme an, wenn ein Wert Null im copyrule.LibraryObject gesetzt ist.

5.18.5 Suchen von Bildregelgruppen und Bildregeln

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- Ein TIA Portal-Projekt mit Bildregeln und Bildregelgruppen ist vorhanden.

Suchen von Bildregelgruppen in Bildgruppen

Fall 1: Suchen von Bildregeln in der Bildregeltabelle.

Mit der folgenden API `sivarc.ScreenRules.Rules` können Sie die verfügbaren Regeln in der SiVArc-Bildregeltabelle wie nachfolgend gezeigt suchen:

//Sammlung sämtlicher direkter Bildregeln der ersten Ebene

```
ScreenRuleComposition screenRules =
sivarc.ScreenRules.Tables.Find("Default screen rule table").Rules;
if (screenRules != null && screenRules.Count > 0)
{
//Suchen von Bildregel
ScreenRule rule = screenRules.Find("Screen rule_7");
if (rule != null)
{
Console.WriteLine("Screen Rule Name: " + rule.Name);
Console.WriteLine("Screen Rule Comment: " + rule.Comment);
Console.WriteLine("Screen Rule Enabled: " + rule.Enabled);
Console.WriteLine("Screen Rule Condition: " + rule.Condition);
}
}
```

Fall 2: Suchen von Bildregelgruppen in der Bildregeltabelle.

Mit der folgenden API `sivarc.ScreenRule.Groups` können Sie die verfügbaren Regeln und Regelgruppen in der SiVArc-Bildregeltabelle wie nachfolgend gezeigt suchen:

```
var groups = sivarc.ScreenRules.Tables.Find("Default screen rule
table").Groups;
if (groups != null && groups.Count > 0)
{
ScreenRule rule = groups.Find("Screen rule_group").Rules.Find("Screen
rule_2");
if (rule != null)
{
Console.WriteLine("Found Screen Rule Name: " + rule.Name);
Console.WriteLine("Found Screen Rule Comment: " + rule.Comment);
Console.WriteLine("Found Screen Rule Enabled: " + rule.Enabled);
Console.WriteLine("Found Screen Rule Condition: " + rule.Condition);
}
var group = groups.Find("Screen rule_group").Groups.Find("Screen rule
group_2");
if(group != null)
{
Console.WriteLine("Found Screen Rule Group Name: " + group.Name);
Console.WriteLine("Found Screen Rule Group Comment: " + group.Comment);
Console.WriteLine("Found Screen Rule Group Enabled: " + group.Enabled);
Console.WriteLine("Found Screen Rule Group Condition: " + group.Condition);
}
}
```

Projektieren der Eigenschaft Condition Operator

Sie können die Eigenschaft Condition Operator der Bildregel in der Bildregeltabelle wie nachfolgend gezeigt projektieren:

```
ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Groups.Find("Screen rule
group_1").Rules.Find("Screen rule_2");

if (screenRule != null)
{
```

```
screenRule.ConditionOperator = ConditionOperator.GreaterThan;
}
```

Die API `screenRuleGroup.ConditionOperator` wird zum Projektieren der Eigenschaft Condition Operator in der Bildregeltabelle verwendet.

- Bei Bildregeln gilt: Wenn Sie auf eine Bildregel zugreifen, bei der die Eigenschaft Condition Operator nicht vorhanden ist, gibt das System den Wert `ConditionOperator.None` zurück.
 - Wenn Sie auf eine Bildregel zugreifen, bei der die Eigenschaft Condition Operator nicht vorhanden ist, gibt das System den Wert `ConditionOperator.None` zurück.
 - Wenn Sie einen anderen Condition Operator-Wert für den oben erwähnten Regeltyp festlegen, zeigt das System eine wiederherstellbare Ausnahme an
- Bei Bildregelgruppen `ConditionOperator.And` gibt das System eine wiederherstellbare Ausnahme zurück.

Projektieren von Layoutfeldern

Die API `GetLayoutFields` wird zum Projektieren von Layoutfeldern im Bildregeleditor verwendet, wie nachfolgend gezeigt:

```
ScreenRule screenRule = sivarcs.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("Screen rule");

{
    var layoutFields = screenRule.GetLayoutFields();
    foreach (var layoutField in layoutFields)
    {
        if (layoutField.Equals("TestLayoutField"))
        {
            screenRule.LayoutField = layoutField;
        }
    }
}
```

`screenRule.LayoutField` schreibt den Wert in das Layoutfeld.

Um Layoutfelder für die Bildregel zu aktualisieren, verwenden Sie den nachfolgenden Code:

```
if (screenRule != null)
{
    ScreenRule screenRule = sivarcs.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("Screen rule group_1");
    IList<string> layoutFields = screenRule?.GetLayoutFields().
    ToList();
    {
```

```

        if (layoutFields ?.Count > 1)
        {
            screenRule.LayoutField = layoutfields[0];
        }
    }
}

```

5.18.6 Zuweisen von Eigenschaften für Regeln und Regelgruppen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- Ein TIA Portal-Projekt mit Bildregeln und Bildregelgruppen ist vorhanden.

Die Attribute der Regeln und Regelgruppen können über Openness festgelegt werden.

Zuweisen von Eigenschaften für Bildregeln

Um Eigenschaften für Bildregeln (z. B. Freigegeben, Name, Kommentar und Bedingung) zuzuweisen, verwenden Sie die folgende API:

```
screenRule.Condition = "StrComp(Block.Name, \"BlockName\")";
```

Um Eigenschaften für Bildregeln zuzuweisen, ist es obligatorisch, die Regel im Regeleditor zu finden. Weitere Informationen zum Suchen der Bildregel finden Sie im Abschnitt Suchen von Bildregelgruppen und Bildregeln (Seite 993)

```

ScreenRule screenRule = sivarcs.ScreenRules.Tables.Find("Default screen
rule table").Rules.Find("Screen rule_1");
if (screenRule != null)
{
    //Bildregelattribute festlegen
    screenRule.Name = "RulenameTest";
    screenRule.Comment = "CommentTest";
    screenRule.Condition = "StrComp(Block.Name, \"BlockName\")";
    screenRule.Enabled = true;
}

```

Zuweisen von Eigenschaften für Bildregelgruppen

Um Eigenschaften für Bildregelgruppen (z. B. Freigegeben, Name, Kommentar und Bedingung) zuzuweisen, verwenden Sie die folgende API:

```
Screenrulegroup.Condition = "StrComp(Block.Name, \"BlockName\")";
```

Um Eigenschaften für Bildregeln zuzuweisen, ist es obligatorisch, die Regel im Regeleditor zu finden. Weitere Informationen zum Suchen der Bildregel finden Sie im Abschnitt Suchen von Bildregeln oder Bildregelgruppen.

Ausnahmefälle

SiVArc zeigt Ausnahmen für die folgenden Fälle an:

- Null/leerer Wert für Name eingegeben
- Doppelte Werte für Feld Name eingegeben
- Wert des Feldes Name überschreitet die Zeichengrenze von 128.
- Name, Kommentar, Bedingung (für ausgenommene Kopierregeln) und Freigegeben

In den folgenden Tabellen sind die für andere Regeltypen verwendeten APIs aufgeführt:

Regeltyp	APIs
Variablenregel/Variablenregelgruppe	TagRule.Condition/TagRuleGroup.Condition
Textliste/Textlistenregelgruppe	TextlistRule.Condition/ TextlistRuleGroup.Condition
Alarmregel/Alarmregelgruppe	AlarmRule.Condition/AlarmRuleGroup.Condition

5.18.7 Erstellen von Regelobjekt und Regelgruppe

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.

Erstellen von Bildregel und Bildregelgruppe

Um eine Bildregel zu erstellen, wird die Methode `Create` in Übereinstimmung mit `ScreenRuleComposition` verwendet. Die API-Methode `Create` auf der Instanz `ScreenRule` wird für das Erstellen von Regeln verwendet. Der folgende Code zeigt, wie Bildregeln erstellt werden:

```
ScreenRule createdScreenRule = sivarcs.ScreenRules.Tables.Find("Default
screen rule table").Rules.Create("CreatedScreenRule");
if (createdScreenRule != null)
{
    //Überprüfung der Eigenschaft CreatedScreenRule
    Console.WriteLine("Created screen rule name: " + createdScreenRule.Name);
}
```

Wenn `CreatedScreenRule` erfolgreich ist, gibt die API `ScreenRule` die neu erstellte Bildregel mit dem bereitgestellten Namen zurück.

Fall 1: Erstellen von Bildregel in einer Regelgruppe

Um eine Bildregel in einer bereits vorhandenen Bildregelgruppe zu erstellen, müssen Sie die Bildgruppe finden, unter der Sie eine Bildregel erstellen wollen.

Mit der folgenden API `sivarc.ScreenRules.Groups.Find` können Sie die Bildregelgruppe in der SiVArc-Bildregeltabelle wie nachfolgend gezeigt suchen:

```
ScreenRule createdScreenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Groups.Find("Screen rule
group_1").Rules.Create("CreatedScreenRuleInUserGroup");
if (createdScreenRule != null)
{
//Überprüfung der Eigenschaft CreatedScreenRule
Console.WriteLine("Created screen rule name: " + createdScreenRule.Name);
}
```

Weitere Informationen zum Erstellen der Bildregelgruppe finden Sie im Abschnitt Suchen von Bildregeln in Bildgruppen.

Die Ausnahmen sind dieselben wie bei Zuweisen von Eigenschaften für Regeln und Regelgruppen. Weitere Informationen zu Ausnahmen finden Sie im Abschnitt Zuweisen von Eigenschaften für Regeln und Regelgruppen (Seite 996).

Hinweis

Für verschiedene Regeltypen werden ähnliche Methoden verwendet. Sie können den Objektnamen durch den Regeltyp ersetzen.

5.18.8 Arbeiten mit Bildregeltabellen und -ordnern

5.18.8.1 Bildregeltabelle erstellen

Hinweis

Die folgenden APIs können in ähnlicher Weise auch für Melderegeln, Kopierregeln, Variablenregeln und Textlistenregeln verwendet werden.

Fall 1: Erstellen einer Bildregeltabelle im Ordner der Bildregeltabellen

Im Ordner der Bildregeltabellen in der Navigation des SiVArc-Projekts können Sie Bildregeltabellen erstellen. Um eine neue Bildregeltabelle zu erstellen, wird die Methode `Create` verwendet.

```
ScreenRuleTable screenRuleTable = sivarc.ScreenRules.Tables.Create
(name: "OpnsScreenRuleTable")
If (screenRuleTable !=null)
{
Console.WriteLine("Created Screen Rule Table : " +
screenRuleTable.Name);
}
```

Fall 2: Hinzufügen einer Bildregeltabelle im vom Benutzer erstellten Ordner

Sie können eine Bildregeltabelle auch im vom Benutzer erstellten Ordner hinzufügen. Die Methode `Create` wird bei `ScreenRuleTableComposition` aufgerufen.

```
ScreenRuleTable screenRuleTable =
sivarc.ScreenRules.Folders.Find(name: "RuleFolder").Tables.Create
(name : "OpnsScreenRuleTable_1")

If (screenRuleTable != null)
{
    Console.WriteLine("Created Screen Rule Table : " +
screenRuleTable.Name);
}
```

5.18.8.2 Bildregelordner erstellen

Fall 1: Erstellen eines Bildregelordners im Ordner der Bildregeltabellen

Im Ordner der Bildregeltabellen in der Navigation des SiVArc-Projekts können Sie einen Bildregelordner erstellen. Um einen neuen Bildregelordner zu erstellen, wird die Methode `Create` verwendet.

```
ScreenRuleFolder screenRuleFolder =
sivarc.ScreenRules.Folders.Create(name:"Folder_ScreenRuleFolder");

If (screenRuleTable != null)
{
    Console.WriteLine ("Created Screen Rule Folder : " +
screenRuleFolder.Name);
}
```

Fall 2: Erstellen eines Ordners im benutzerdefinierten Ordner

Sie können einen Ordner im benutzerdefinierten Ordner unter Bildregeln erstellen.

```
ScreenRuleFolder screenRuleFolder =
sivarc.ScreenRules.Folders.Find (name: "RuleFolder").Folders.Create
(name: "Folder_ScreenRuleFolder_1")
{
    Console.WriteLine("Created Screen Rule Folder : " +
screenRuleFolder.Name);
}
```

5.18.8.3 Zugreifen auf Bildregeln und -gruppen

Um auf die Bildregeln und -gruppen der `RootScreenRuleTable` zuzugreifen (vom Benutzer erstellte Tabelle mit Regeln und Regelgruppen), verwenden Sie den folgenden Code:

```
Console.WriteLine("Rules");
```

```
var rules =
sivarc.ScreenRules.Tables.Find("RootScreenRuleTable").Rules;

if (rules != null && rules.Count > 0)
{
    foreach (var rule in rules)
    {
        Console.WriteLine(rule.Name);
    }
}

Console.WriteLine("Groups");
var groups =
sivarc.ScreenRules.Tables.Find("RootScreenRuleTable").Groups;

if (groups != null && groups.Count > 0)
{
    foreach (var group in groups)
    {
        Console.WriteLine(group.Name);
    }
}
```

Um auf die Regeln und Regelgruppen der Regeltabelle im Ordner zuzugreifen, verwenden Sie den folgenden Code:

```
Console.WriteLine("Rules");
var rules =
sivarc.ScreenRules.Folders.Find("RuleFolder").Tables.Find("AddedScreenRuleTable").Rules;
if (rules != null && rules.Count > 0)
{
    foreach (var rule in rules)
    {
        Console.WriteLine(rule.Name);
    }
}

Console.WriteLine("Groups");
var groups =
sivarc.ScreenRules.Folders.Find("RuleFolder").Tables.Find("AddedScreenRuleTable").Groups;
if (groups != null && groups.Count > 0)
{
    foreach (var group in groups)
    {
        Console.WriteLine(group.Name);
    }
}
```

5.18.8.4 Löschen der Bildregeltabelle und des benutzerdefinierten Regelordners

Um RootScreenRuleTable im Ordner der Bildregeltabellen zu löschen, verwenden Sie folgendes Code-Snippet:

```
sivarc.ScreenRules.Tables.Find(name: "RootScreenRuleTable").Delete();
```

Um den benutzerdefinierten Regelordner zu löschen, verwenden Sie folgendes Code-Snippet:

```
sivarc.ScreenRules.Folders.Find(name: "RuleFolder").Delete();
```

Ausnahme:

Wenn der Benutzer versucht, die Default-Bildregeltabelle wie unten erwähnt zu löschen:

```
sivarc.ScreenRules.Tables.Find(name: "Default screen rule  
table").Delete();
```

Das System gibt eine wiederherstellbare Ausnahme zurück mit der Meldung, dass die Default-Regeltabelle nicht gelöscht werden kann.

5.18.8.5 Auf die Regeltabelle in Library Type zugreifen

Für den Zugriff auf die Regeltabelle im Ordner Library Type wird folgende API verwendet:

```
LibraryType libraryType =  
project.ProjectLibrary.TypeFolder.Types.Find (name:"Screen rule  
table")
```

Zum Iterieren durch verschiedene Versionen in Library Type wird folgende API verwendet:

```
LibraryType libraryType =  
project.ProjectLibrary.TypeFolder.Types.Find (name:"Screen rule  
table")  
foreach (var version in LibraryType.Versions)  
Console.WriteLine ("Library type {0} version is {1}",  
libraryType.Name, version.VersionNumber);
```

5.18.8.6 Instanz einer Regeltabelle erstellen

Um eine Instanz einer Regeltabelle von Library Type in PNV zu erstellen, wird die Methode `CreateFrom` verwendet. Die Methode `CreateFrom` verwendet die Zusammensetzung der Regeltabelle zum Instanzieren einer spezifischen Regeltabellenversion in der Projektnavigation. `TypeVersion` muss für die entsprechende Regeltabelle umgewandelt werden. Der folgende Code erläutert das:

```
LibraryType libraryType =  
project.ProjectLibrary.TypeFolder.Types.Find (name:"Screen rule  
table");  
Version version = new Version ("0.0.2");  
var typeVersion = LibraryType.Versions.Find(version);  
ruleTableComp.CreateFrom(typeVersion as ScreenRuleTableTypeVersion)
```

Hinweis

Werden Bearbeitungsvorgänge mit der API im Freigabemodus versucht, gibt die SiVArc-Regeltabelle eine Ausnahme zurück.

Versionsdetails der instanziierten Regeltabelle

Um die Versionsdetails der Regeltabelle zu finden, die in der Projektnavigation instanziiert ist, wird die Methode `LibraryTypeVersion` wie nachfolgend gezeigt verwendet:

```
var ruleTableComp = sivarcs.ScreenRules.Tables;  
ScreenRuleTable screenRuleTable = ruleTableComp.Find(name: "Screen  
rule table");  
LibraryTypeInfo instanceInfo =  
screenRuleTable.GetService<LibraryTypeInfo>();  
LibraryTypeVersion connectedversion =  
instanceInfo.LibraryTypeVersion;
```

5.18.9 Auf Ordner unter "Kopiervorlage/Typ eines Bildes" zugreifen

Auf Ordner in Library Type zugreifen

Um auf einen beliebigen Ordner Library Type innerhalb der Kopiervorlage zuzugreifen, wird die API `LibraryTypeFolder` wie unten gezeigt verwendet:

```
LibraryTypeFolder libraryTypeFolder =  
project.ProjectLibrary.TypeFolder.Folders.Find("Folder");  
  
if (libraryTypeFolder != null && screenRule != null)  
{  
    screenRule.LibraryScreen = libraryTypeFolder;  
}
```

Auf Ordner in der Kopiervorlage zugreifen

Der Ordner Library Type wird der Kopiervorlage wie unten gezeigt zugewiesen:

```

MasterCopyFolder masterCopyFolder =
project.ProjectLibrary.MasterCopyFolder.Folders.Find("LegacyItems");
if (masterCopyFolder != null && screenRule != null)
{
    screenRule.LibraryScreen = masterCopyFolder;
}

```

Hinweis

- Wenn Sie den Wert von Kopiervorlage/Typ eines Bildobjekts als Ordner festlegen und dann versuchen, Bildobjekt-/Layout-Spalten festzulegen, gibt das System eine Ausnahme zurück.
 - Wenn Sie nach dem Festlegen des Werts für die Bildobjektspalte versuchen, unter Kopiervorlage/Typ eines Bilds nach einem Ordner zu suchen, löscht SiVArc die bereits konfigurierten Werte in den Spalten mit den Bildobjekt- und Layout-Feldern.
-

5.18.10 Zugreifen auf Bibliotheksobjekte

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- Erstellen oder Verwenden von bereits vorhandenen Bildregeln
- Ein TIA Portal-Projekt mit Kopiervorlagen von Bildern, Bildobjekten, Programmbausteinen und Bildbausteintypen ist vorhanden.

Lesen und Schreiben von Programmbausteineigenschaften

Fall 1: Betrachten wir das Lesen des Programmbausteinattributs einer Bildregel. Diese können projiziert werden von:

- PLC-Baustein
- Bibliothekstyp
- Kopiervorlage der Bibliothek

Der folgende Code zeigt, wie das Programmbausteinattribut der Bildregel aus der PLC gelesen wird:

```

if (sivarc != null)
{
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("ScreenRule")

    ISivarcProgramBlockSource programblock = screenRule?.ProgramBlock;
    if (programBlock != null)

```

```

    {
        CodeBlock plcBlock = programBlock as CodeBlock;
        Console.WriteLine ("Screen Rule program block: " +
plcBlock?.Name);
    }
}

```

- Wenn Sie den Programmbausteinwert aus dem Bibliothekstyp lesen, stellen Sie sicher, dass Sie `programBlock` in `LibraryType` typisieren
`LibraryType libraryType = programBlock as LibraryType`
- Wenn Sie den Programmbausteinwert aus der Kopiervorlage der Bibliothek lesen, stellen Sie sicher, dass Sie `programBlock` in `MasterCopy` typisieren
`MasterCopy masterCopy = programBlock as MasterCopy`

Fall 2: Betrachten wir das Schreiben des Programmbausteinattributs einer Bildregel. Sie können den Wert für einen Programmbaustein aus Folgendem schreiben

- PLC-Baustein
- Bibliothekstyp
- Kopiervorlage der Bibliothek

Der folgende Code zeigt, wie das Programmbausteinattribut der Bildregel aus der PLC geschrieben wird:

```

if (sivarc != null)
{
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("ScreenRule");
    Device dev = myProject.Devices.Find("S71500/ET200MP station_1");
    softwareContainer softwareContainer = null
    var firstorDefault = dev.DeviceItems.FirstorDefault(name =>
name.Name.Equals("PLC_1"));
    if (firstorDefault != null)
    SoftwareContainer = firstorDefault.GetService<SoftwareContainer>();
    if (softwareContainer != null)
    {
        Software softTarget = softwareContainer.Software;
        PLCSoftware plcSoftware = (PLCSoftware) softTarget;
        var userBlocks = plcSoftware.BlockGroup.Blocks;
        screenRule.ProgramBlock=userBlocks.Find ("Block_2") as CodeBlock;
    }
}

```



```
}
```

- Wenn Sie den Programmbausteinwert aus dem Bibliothekstyp schreiben:

```
screenRule.ProgramBlock =  
project.ProjectLibrary.TypeFolder.Types.Find ("Block_Type");
```

- Wenn Sie den Programmbausteinwert aus der Kopiervorlage der Bibliothek schreiben:

```
screenRule.ProgramBlock =  
project.ProjectLibrary.MasterCopyFolder.MasterCopies.Find  
("Block_2");
```

Hinweis

Das Attribut `ProgramBlock` wird auch für andere unterstützte Editoren geändert.

Lesen und Schreiben der Eigenschaft `ScreenObjectLibraryItem` einer Bildregel

Fall 1: Betrachten wir das Lesen des Attributwertes `ScreenObjectLibraryItem` einer Bildregel. Sie können den Wert für ein Bibliothekselement des Bildobjekts aus der Kopiervorlage wie folgt lesen:

```
if (sivarc != null)  
{  
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default  
screen rule table").Rules.Find("ScreenRule")  
  
    ISivarcLibraryItem screenObjectMasterCopy =  
screenRule?.ScreenObjectLibraryItem;  
    if (screenObjectMasterCopy != null)  
    {  
        MasterCopy masterCopy= screenObjectMasterCopy as MasterCopy;  
        Console.WriteLine ("Screen object master copy name: " +  
masterCopy?.Name);  
    }  
}
```

- Wenn Sie das Bildobjekt aus dem Bibliothekstyp lesen, stellen Sie sicher, dass Sie `screenObjectMasterCopy` in `LibraryType` typisieren
`LibraryType libraryType = screenObjectMasterCopy as LibraryType;`

Fall 2: Betrachten wir das Schreiben des Attributwertes `ScreenObjectLibraryItem` einer Bildregel. Sie können den Wert aus der Bibliothek wie folgt schreiben:

```
if (sivarc != null)  
{  
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default  
screen rule table").Rules.Find("ScreenRule");
```

```

    screenRule.ScreenObjectLibraryItem =
project.ProjectLibrary.MasterCopyFolder.MasterCopies.Find
("Button_Lib");
}

```

- Wenn Sie den Bildobjektwert aus dem Bibliothekstyp schreiben:

```

screenRule.ScreenObjectLibraryItem
=project.ProjectLibrary.TypeFolder.Types.Find ("Faceplate_Type");

```

Hinweis

Auf andere Attribute des Kopierregel-Bibliotheksobjekts (`LibraryItem`) vom Typ `ISivarcLibraryItem` kann zugegriffen werden, um den Wert auf ähnliche Weise zu lesen/schreiben.

Lesen und Schreiben der Bild-Master-Eigenschaft einer Bildregel

Fall 1: Betrachten wir das Lesen des Attributwertes `LibraryScreen` einer Bildregel. Sie können den Wert für ein Bibliothekselement des Bildes aus der Kopiervorlage wie folgt lesen:

```

if (sivarc != null)
{
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("ScreenRule")

    ISivarcLibraryMasterCopy screenLibraryMasterCopy =
screenRule?.LibraryScreen;

    if (screenLibraryMasterCopy != null)
    {
        MasterCopy masterCopy= screenLibraryMasterCopy as MasterCopy;
        Console.WriteLine ("Screen master copy name: " +
mastercopy?.Name);
    }
}

```

Fall 2: Betrachten wir das Schreiben des Attributwertes `LibraryScreen` einer Bildregel. Sie können den Wert für die Kopiervorlage des Bildes der Bildregel aus der Kopiervorlage der Bibliothek wie folgt schreiben:

```

if (sivarc != null)
{
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("ScreenRule");

    screenRule.LibraryScreen =
project.ProjectLibrary.MasterCopyFolder.MasterCopies.Find
(Screen_lib);
}

```

}

Hinweis

Auf andere Attribute vom Typ `ISivarcLibraryMasterCopy` (`TextlistLibraryItem` und `AlarmLibraryItem`) kann zum Lesen/Schreiben des Wertes auf ähnliche Weise zugegriffen werden.

Das Attribut `LibraryScreen` wird genutzt, um Kopiervorlage und Typ eines Bilds zu lesen und zu bearbeiten. Mit dem `LibraryScreen`-Attribut können Sie in der Spalte **Master copy/ Type of screen** entweder nach der Kopiervorlage oder nach dem Typ suchen. Folgender Code zeigt die Navigation unter `project > project Library > Typefolder > with name "<<screen_type>>"`. Das `LibraryScreen`-Attribut wird dem Wert des Bildtyps zugewiesen:

```
Sivarc sivarc = project.GetService<Sivarc>();
if (sivarc != null)
{
    ScreenRules screenRuleEditor = sivarc.ScreenRules;
    if (screenRuleEditor != null)
    {
        ScreenRuleComposition screenRuleComposition =
            sivarc.ScreenRules.Tables.Find("Default screen rule
table").Rules;
        if (screenRuleComposition != null && screenRuleComposition.Count > 0)
        {
            ScreenRule scRule = screenRuleComposition.Find("Screen rule");
            if (scRule != null)
            {
                Console.WriteLine((scRule.LibraryScreen as MasterCopy)?.Name);
                scRule.LibraryScreen =
                    project.ProjectLibrary.TypeFolder.Types.Find("Screen_Type");
                Console.WriteLine((scRule.LibraryScreen as LibraryType)?.Name);
            }
        }
    }
}
```

5.18.11 Projektieren des PLC- und HMI-Geräts

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- Ein TIA Portal-Projekt mit HMI-Geräten, Panel-Geräten und PLCs ist vorhanden.

Lesen und Schreiben eines PLC/HMI-Geräts

Dynamische Eigenschaften bestehen ausschließlich aus PLC-Gerätespalten und HMI-Gerätespalten.

Fall 1: Betrachten wir das Lesen dynamischer Eigenschaften für ein Regelobjekt. Sie können Attribute wie den Status des PLC- und HMI-Geräts wie im folgenden Code gezeigt lesen:

Um den Status des PLC-Geräts zu überprüfen, sofern für eine bestimmte Regel ausgewählt, müssen Sie den PLC-Namen an die Methode `GetAttribute` übergeben.

```
if (sivarc != null)
{
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("ScreenRule")
    var plcStatus = screenRule.GetAttribute("PLC_1");
    Console.WriteLine ("PLC_1 status: " + plcStatus);
}
```

Um den Status des HMI-Geräts zu überprüfen, sofern für eine bestimmte Regel ausgewählt, müssen Sie den Runtime-Namen des HMI-Geräts an die Methode `GetAttribute` übergeben.

Bei Panels müssen Sie nur den Gerätenamen übergeben.

```
var hmiDeviceStatus = screenRule.GetAttribute("HMI_RT_2");
Console.WriteLine ("HMI_RT_2 status: " + HMIDeviceStatus);
```

Um den Status für alle Geräte zu lesen, gehen Sie folgendermaßen vor:

```
var devicestatus = screenRule.GetAttribute new List<string>
{
    "PLC_1", "PLC_2", "HMI_RT_1", "HMI_RT_2";
}
foreach (var devicestatus in devicestatus)
{
    Console.WriteLine ("PLC & HMI device status: " + devicestatus);
}
}
```

Fall 2: Betrachten wir das Schreiben dynamischer Eigenschaften für ein Regelobjekt. Sie können folgendermaßen Attribute wie PLC-Gerätespalten lesen, um den Status des PLC-Geräts zu überprüfen, sofern für eine Regel ausgewählt:

Um den Status des PLC-Geräts zu aktualisieren, sofern für eine bestimmte Regel ausgewählt, stellen Sie sicher, dass Sie den PLC-Namen an die Methode `SetAttribute` mit einem booleschen Wert übergeben.

```
if (sivarc != null)
{
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("ScreenRule")
    screenRule.SetAttribute("PLC_1", false);
}
```

Um den Status des HMI-Geräts zu aktualisieren, sofern für eine bestimmte Regel ausgewählt, stellen Sie sicher, dass Sie den Runtime-Namen des HMI-Geräts an die Methode `SetAttribute` mit einem boolschen Wert übergeben.

Bei Panels müssen Sie nur den Gerätenamen übergeben.

```
if (sivarc != null)
{
    ScreenRule screenRule = sivarc.ScreenRules.Tables.Find("Default
screen rule table").Rules.Find("ScreenRule");
    screenRule.SetAttribute("HMI_RT_1", true);
}
```

Um den Status für alle Geräte zu aktualisieren, gehen Sie folgendermaßen vor:

```
if (sivarc != null)
{
    var valuesToBeUpdate = new[]
    {
        new KeyValuePair<string, object>("PLC_2", true),
        new KeyValuePair<string, object>("PLC_3", true),
        new KeyValuePair<string, object>("HMI_RT_1", false),
        new KeyValuePair<string, object>("HMI_1", true),
    };
    ScreenRule screenRule =
sivarc.ScreenRules.Rules.Find("ScreenRule");
    screenRule.SetAttributes(valuesToBeUpdate);
}
```

Ausnahmefälle

- **Wiederherstellbare Ausnahme:** Beim Zugriff auf PLC/HMI-Gerätespalten über `GetAttributes` für ein anderes Regelobjekt, bei dem dynamische Attribute nicht unterstützt werden (Variablen- und Textlistenregel), zeigt das System eine wiederherstellbare Ausnahme an, z. B. `{'HMI_RT_1' wird vom Typ 'Siemens.Engineering.SiVArc.TagRule' nicht unterstützt}`.
- **Nicht wiederherstellbare Ausnahme** – Bei ungültigen PLC/HMI-Gerätenamen, die nicht auf ein Regelobjekt anwendbar sind, zeigt das System eine nicht wiederherstellbare (NonRecoverable) Ausnahme an; dies gilt nur für Bild-, Melde- und Kopierregeln.

Hinweis

Bei Kopierregeln können Sie den PLC-Status abfragen, obwohl die PLC-Spalte nicht unterstützt wird. Grund dafür ist das allgemeine Verhalten von Editoren, die Funktionalität von SiVArc wird dadurch nicht beeinflusst.

Für den Zugriff auf den PLC-/HMI-Gerätestatus für Regel-/Regelgruppenobjekte wird das folgende dynamische Attribut verwendet: `object status = screenRule.GetAttribute("PLC device name");`

Einschränkungen:

- HMI-Gerätenamen mit alphanumerischen Zeichen dürfen keine Sonderzeichen enthalten, außer ' _ '
- Um den HMI-Gerätestatus abzufragen, sind die Namen mit "HMI_" als Präfix, gefolgt vom aufgelösten Gerätenamen anzugeben. Beispiel: HMI-Gerätename: HMI_@#\$RT_1, Für den Zugriff auf den Gerätenamen müssen Sie Parameter HMI_HMI_RT_1 angeben.
- Projekt mit mehreren HMI-Geräten mit nur "_" als Sonderzeichen wird in den gleichen Gerätenamen aufgelöst, das System hängt die Zahl zusammen mit dem HMI-Gerätenamen an. Beispiel: Device_PLC_XYZ steht für das erste Gerät in der Projekthierarchie. Die gleiche Vorgehensweise gilt für den Zugriff auf PLC-Gerätestatusdaten.
- Projekt mit zwei Geräten – HMI_@#RT_1 & HMI_%^RT_@#1; wenn beide in den gleichen Namen aufgelöst werden, also HMI_RT_1 nach Entfernen der Sonderzeichen (Projekt HMI_@#RT_1 HMI_%^RT_@#). Für den Zugriff auf den Status des ersten Geräts müssen Sie HMI_HMI_RT_1 als Parameter angeben, für den Zugriff auf den Status des zweiten Geräts müssen Sie HMI_HMI_RT_1_1 als Parameter angeben. Die gleiche Vorgehensweise gilt für den Zugriff auf PLC-Gerätestatusdaten.

5.18.12 Löschen von Regeln und Regelgruppen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- Ein TIA Portal-Projekt mit Bildregeln und Bildregelgruppen ist vorhanden.

Löschen von Regeln und Regelgruppen

Um eine Bildregelgruppe zu löschen, verwenden Sie die folgende API:

```
sivarc.ScreenRules.Tables.Find("Default screen rule table").Rules.Find("Screen rule_1").Delete();
```

`ScreenRules` ist ein Ankerobjekt für alle Bildregelobjekte. Um eine Löschung durchzuführen, ist es obligatorisch, die Regel im Regeleditor zu finden. Weitere Informationen zum Suchen der Bildregel finden Sie im Abschnitt "Suchen von Bildregeln oder Bildregelgruppen".

Um Bilder aus einer Bildgruppe zu löschen, verwenden Sie die folgende API:

```
sivarc.ScreenRules.Tables.Find("Default screen rule
table").Rules.Find("Screen rule group_1").Delete();
```

5.18.13 Projektieren von Variablenregeln

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- Ein TIA Portal-Projekt mit Variablenregeln und Variablenregelgruppen ist vorhanden.

Zuweisen von Eigenschaften für Variablenregeln

Sie können Eigenschaften für Variablenregeln (z. B. Name, Bedingung, Kommentar und Freigegeben) zuweisen. Die API `TagRule` wird zum Einstellen von Eigenschaften für Variablenregeln verwendet, wie nachfolgend gezeigt:

```
TagRule tagrule = sivarc.TagRules.Tables.Find("Default tag rule
table").Rules.Find ("Tag rule_1");
if (tagRule != null)
{
    tagRule.Name = "TagRuleName" ;
    tagRule.Comment = "TagRuleComment";
    tagRule.Condition = "StrComp(Block.Name, \"BlockName\")";
    tagRule.Enabled = true;
}
```

Die API `TagRuleGroup` wird auf ähnliche Weise zum Festlegen der Eigenschaften der Variablenregelgruppen verwendet.

Hinweis

Wenn der für die Eigenschaften Bedingung und Kommentar übergebene Wert 500 Zeichen überschreitet, wird eine Ausnahme angezeigt.

Um zusätzliche Eigenschaften der Variablenregeln wie `TagGroupHierarchy`, `TagTable` und `ConditionalOperator` zu aktualisieren, verwenden Sie den folgenden API-Code:

Die APIs `TagGroupHierarchy`, `TagTable` und `ConditionOperator` werden zum Ändern der Variablengruppen in Hierarchie, Variablentabelle und des Condition Operator, der für eine Variablenregel im Variablenregeleditor zugewiesen ist, wie folgt verwendet:

```
TagRule tagRule = sivarc.TagRules.Tables.Find("Default tag rule
table").Groups.Find("Tag rule group_1").Rules.Find("Tag rule_2");
if (tagRule != null)
{
```

```
tagRule.TagGroupHierarchy = "HmiTag.DB.FolderPath";  
tagRule.TagTable = "Hmitag.DB.SymbolicName";  
tagRule.ConditionOperator = ConditonOperator.Equal;  
}
```

Ausnahmefälle

SiVArc zeigt Ausnahmen für die folgenden Fälle an:

- Null/leerer Wert wurde für `TagTable` eingegeben
- Doppelte Werte für Feld Name eingegeben
- Der Wert der Felder `TagGroupHierarchy` und `TagTable` überschreitet die Zeichengrenze von 500.
- Für ungültige Werte von `ConditionOperator` wird `ConditionOperator.None` zurückgegeben

Siehe auch

Zuweisen von Eigenschaften für Regeln und Regelgruppen (Seite 996)

5.18.14 Variablen-/Textdefinitionen konfigurieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- PLC mit dem zugehörigen Baustein

Variablendefinitionen erstellen

Um eine Variablendefinition für einen Baustein zu erstellen, müssen Sie zuerst die PLC gefolgt vom zugehörigen Baustein identifizieren und dann auf den SiVArc-Datenprovider für diesen Baustein zugreifen. Sobald Sie auf den SiVArc-Datenprovider zugreifen können, erstellen

Sie mit `TagDefinitionComposition` eine neue Variablendefinition. Das folgende Code-Snippet zeigt dies an:

```
CodeBlock block = plcSoftware.BlockGroup.Blocks.Find("Block_1") as
CodeBlock;
// Zugriff auf Dienst SivarcDataProvider
SivarcDataProvider sivarcDataProvider =
block.GetService<SivarcDataProvider>();
// Eintrag für Variablendefinition erstellen
TagDefinition tagDefinition =
sivarcDataProvider.TagDefinitions.Create("Tag_definition_1");
if (tagDefinition != null)
{
tagDefinition.Value = "Tag_Definition_1_Value";
tagDefinition.Comment = "Tag_Definition_1_Comment";
}
```

Zugriff auf Textdefinitionen

Um auf Textdefinitionen für einen Baustein zuzugreifen, müssen Sie zuerst die PLC gefolgt vom zugehörigen Baustein identifizieren und dann auf den SiVArc-Datenprovider für diesen Baustein zugreifen. Wenn Sie Zugriff auf den SiVArc-Datenprovider erhalten haben, legen Sie mit `TextDefinitionComposition` fest, für welche Textdefinitionsspalte die Werte geschrieben werden sollen.

Das folgende Code-Snippet zeigt dies an:

```
TextDefinitionComposition textDefinitions =
sivarcDataProvider.TextDefinitions;
TextDefinition textDefinition = textDefinitions.Find("Text_definition");
if (textDefinition != null)
{
textDefinition.Name = "Updated_Text_definition";
textDefinition.Expression = "Block.DB.SymbolicName";
textDefinition.Comment = "Updated_Text_definition_Comment";
var text = textDefinition.Text;
//Deutschen Text aktualisieren
Language germanLanguage = project.LanguageSettings.Languages.Find(new
CultureInfo("de-DE"));
text.Items.Find(germanLanguage).Text = "GermanText";
}
```

Für den Lesezugriff können Sie die gleichen Schritte ausführen.

Sie können auch auf den konfigurierten Wert für Textdefinitionen zugreifen, die mehrsprachige Anwendungen unterstützen. Wählen Sie mit `"project.LanguageSettings.Languages.Find("<Name_der_Sprache>")"` die Sprache, für die Sie Werte lesen/schreiben möchten.

Objekt `MultilingualTextItem` enthält den Wert für den Sprachtyp.

Variablen-/Textdefinitionen löschen

Um Variablendefinitionen für einen Baustein zu löschen, müssen Sie zuerst die PLC gefolgt vom zugehörigen Baustein identifizieren und dann auf den SiVArc-Datenprovider für diesen Baustein zugreifen. Sobald Sie auf den SiVArc-Datenprovider zugreifen können, suchen Sie die Variablendefinition, die Sie mit `tagDefinition.Delete` löschen möchten.

Die gleiche Vorgehensweise gilt für Textdefinitionen, für die Löschung wird die API `textDefinitions.Delete` verwendet.

Das folgende Code-Snippet zeigt dies an:

```
string projectPath = @"D:\Temp\Project\Project.ap19";
Project project = portal.Projects.Open(new
System.IO.FileInfo(projectPath));
Device myDevice = null;
foreach (var device in project.Devices)
{
//Gerät suchen
if (!device.Name.Equals("S71500/200 station_1"))
{
continue;
}
myDevice = device;
}
if (myDevice?.DeviceItems != null)
{
foreach (var deviceItem in myDevice.DeviceItems)
{
//PLC suchen
if (deviceItem.Name == "PLC_1")
{
SoftwareContainer container = deviceItem.GetService<SoftwareContainer>();
Software target = container.Software;
PlcSoftware software = (PlcSoftware) target;
//Programmbaustein suchen
CodeBlock block = software.BlockGroup.Blocks.Find("Block_1") as CodeBlock;
// Zugriff auf Dienst SivarcDataProvider
SivarcDataProvider sivarcDataProvider =
block.GetService<SivarcDataProvider>();
// Variablendefinition löschen
sivarcDataProvider.TagDefinitions.Find("Tag_definition").Delete();
// Textdefinition löschen
sivarcDataProvider.TextDefinitions.Find("Text_definition").Delete();
}
}
}
```

5.18.15 UMAC-Einrichtung für Openness

Wissenswertes zu UMAC

Die Openness-Autorisierung wird für jede Änderung verwendet und von SiVArc wird keine zusätzliche UMAC-Validierung vorgenommen. Bei nicht autorisierten Benutzern gibt das System die wiederherstellbare Ausnahme 'FunctionRightNotFoundException' zurück.

5.18.16 SiVArc-Generierung

Voraussetzung

- Starten Sie die TIA Portal Openness-Anwendung. Weitere Informationen zu Verbindungen finden Sie im Benutzerhandbuch zum TIA Portal.
- Ein mit einem HMI-Gerät verbundenes TIA Portal-Projekt ist vorhanden und der PLC konfiguriert.

Wichtige Hinweise

- Stellen Sie sicher, dass die SiVArc-Lizenz auf Ihrem PC installiert ist, andernfalls wird während der Generierung eine Ausnahme gemeldet - *"SiVArc-Lizenz fehlt; eine SiVArc-Lizenz ist für die Änderung von Daten obligatorisch"*.
- Stellen Sie sicher, dass ein gültiger Geräte-Name verwendet wird, andernfalls wird eine Ausnahme gemeldet - *"HMI-Gerät 'Gerätename' nicht gefunden"*.
- Stellen Sie sicher, dass ein gültiger PLC-Name aufgerufen wird, andernfalls wird eine Ausnahme gemeldet - *"PLC-Gerät 'PLC-Gerätename' nicht gefunden"*.
- Stellen Sie sicher, dass der Name eines unterstützten Geräts verwendet wird, andernfalls wird eine Ausnahme gemeldet - *"HMI-Gerät 'Gerätename' wird nicht unterstützt"*.
- Stellen Sie sicher, dass der Name eines unterstützten PLC verwendet wird, andernfalls wird eine Ausnahme gemeldet - *"PLC-Gerät 'PLC-Gerätename' wird nicht unterstützt"*.
- Stellen Sie sicher, dass Sie einen gültigen Parameter GenerationOption übergeben. Wird kein Parameter übergeben, erfolgt die SiVArc-Generierung, und standardmäßig werden dafür die TIA Portal-Projekteinstellungen verwendet.
- Stellen Sie sicher, dass Sie einen gültigen PLC-Namen verwenden, der nicht bei der vorherigen Generierung verwendet wurde, da ansonsten das System einfriert.

Generierungsoptionen - Enum (Merker)

SiVArc unterstützt Enum-Optionen, und Sie können in der API `Generate` eine Kombination aus zwei Werten übergeben. Die folgende Tabelle zeigt die Enum-Optionen:

SN	Werte	Beschreibung
1	None	Keine Auswahl getroffen, für die Generierung werden die Standardeinstellungen übernommen
2	AllTags	Es werden alle Variablen generiert
3	UsedHmi- Tags	Es werden nur relevante (verwendete) Variablen generiert

4	FullGenerati-on	Wenn die Option FullGeneration nicht ausgewählt ist, entscheidet SiVArc intern basierend auf der Konfiguration, ob eine vollständige Generierung oder eine Delta-Generierung durchgeführt werden muss. Wenn Sie den Parameter FullGeneration übergeben, führt SiVArc eine erzwungene vollständige Generierung durch.
5	UserCreated-Rules	Nur vom Benutzer erstellte Regeln werden ausgeführt
6	EnergySuite-Rules	Nur EnergySuite-Regeln werden ausgeführt
7	AllRules	Alle Regeln werden ausgeführt

Um SiVArc zu generieren, verwenden Sie die folgende API:

```
sivarc.Generate("HMI_1", new List<string> {PLC_1},
GenerateOptions.AllTags | GenerateOptions.FullGeneration);
```

SiVArcGenerationResult und SivarcFeedbackMessage

Bei der SiVArc-Generierung wird bei erfolgreicher Generierung auf die folgenden Eigenschaften zugegriffen:

- **IsGenerationSuccessful** - Informiert darüber, ob die SiVArc-Generierung erfolgreich ist.
- **WarningCount** - Gesamtzahl von Warnungen nach der SiVArc-Generierung
- **ErrorCount** - Gesamtzahl der Fehler nach der SiVArc-Generierung
- **Messages** - Zusammensetzung der Rückmeldung

Um das SiVArc-Ergebnis zu generieren, verwenden Sie die folgende API:

```
private void WriteSivarcGenerationResults(SivarcGenerationResult result)
{
    Console.WriteLine("Is SiVArc generation successful:" +
        result.IsGenerationSuccessful);
    Console.WriteLine("Tatal warning count:" + result.WarningCount);
    Console.WriteLine("Total error count:" + result.ErrorCount);
    RecursivelyWriteMessages(result.Messages);
}
```

Bei der SiVArc-Generierung wird bei erfolgreicher Generierung auf die folgenden Rückmeldungen zugegriffen:

- **Path**: Header-Text der Rückmeldung (Header-Meldungen haben immer ein leeres Beschreibungsfeld)
- **DateTime**: Datum und Uhrzeit der Rückmeldung
- **MessageType**: Typ der Rückmeldung
- **Description**: Beschreibung/Inhalt der Rückmeldung (nur wenn der Pfad leer ist, um sicherzustellen, dass es sich nicht um eine Header-Meldung handelt)
- **WarningCount**: Anzahl der Warnungen für eine Header-Meldung

- **ErrorCount:** Anzahl der Fehler für eine Header-Meldung
- **Messages:** Zusammensetzung der Rückmeldung (`SivarcFeedbackMessage`)

Mit dem folgenden Code-Snippet können Sie rekursive Rückmeldungen anzeigen:

```
private void RecursivelyWriteMessages(SivarcFeedbackMessageComposition
messages)
{
    foreach (SivarcFeedbackMessage message in messages)
    {
        Console.WriteLine("Path: " + message.Path);
        Console.WriteLine("DateTime: " + message.DateTime);
        Console.WriteLine("State: " + message.MessageType);
        Console.WriteLine("Description: " + message.Description);
        Console.WriteLine("Warning Count: " + message.WarningCount);
        Console.WriteLine("Error Count: " + message.ErrorCount);
        RecursivelyWriteMessages(message.Messages);
    }
}
```

SiVArc-Generierung für mehrere HMI-Geräte

Mit der API `Generate` können Sie die SiVArc-Generierung ausführen, wenn mehrere HMI-Geräte mit mehreren PLCs mit projektierten Alarmregeln, Variablenregeln, Textlistenregeln, Kopierregeln, Bildregeln verbunden sind. Die API-Methode `Generate` besteht aus drei Argumenten:

- **Devices** - Sammlung der Geräte, für die die SiVArc-Generierung ausgelöst wird
- **PLCs** - Liste der für die SiVArc-Generierung erforderlichen PLCs
- **Generate Options** - Liste der Einstellungen für die SiVArc-Generierung

Der folgende Code verdeutlicht die Methode `Generate`, die für die Generierung mit mehreren projektierten HMI-Geräten und PLCs verwendet wird:

```
if (sivarc != null)
{
    SivarcGenerationResult result = sivarc.Generate(
        new List<string> {"HMI_RT_1", "HMI_RT_2"},
        new List<string> {"PLC_1", "PLC_2"}, GenerateOptions.AllTags |
        GenerationOptions.FullGeneration);
    WriteGenerationResults(result);
}
```

- Bei der `SivarcGeneration` API legt das System bei Ausführung mit `GenerationOptions.none` "All Rules" als Default-Regelsatz fest, wenn kein Regelsatz ausgewählt ist, was bedeutet, dass für das ausgewählte Gerät keine SiVArc-Generierung durchgeführt wird und HMI-Objekte für alle Regeln generiert werden.

```
SivarcGenerationResult result = sivarc.Generate(device  
name: "HMI_RT_1", plcs: new List<string>() {"PLC_1"},  
GenerationOptions.none);
```

- Bei der SivarcGeneration API betrachtet das System bei Ausführung mit "User created rules" das als Regelsatz und generiert entsprechende Bilder mit Objekten.

Hinweis

- Wenn "All Rules" mit "User created Rules" oder "Energy Suite Rules" ausgewählt ist, gibt das System während der Generierung "All Rules" den Vorrang und führt das als Regelsatz aus.
 - Wenn "User created Rules" mit "Energy Suite Rules" ausgewählt ist, gibt das System während der Generierung "User created Rules" den Vorrang und führt das als Regelsatz aus.
-

Ausnahme- und Sonderfälle

- Wenn die API `Generate` mit dem Parameter Null aufgerufen wird, wird eine Ausnahme gemeldet - *"Das Argument 'Argumentname' darf nicht null sein"*.
- Wenn ein beliebiges HMI-Panel/Gerät während der Übergabe der PLC- und HMI-Geräte an die API `Generate` nicht mit der PLC verbunden ist, wird eine Ausnahme gemeldet - *"Das HMI-Panel/Gerät '{0}' ist nicht mit den ausgewählten PLC-Geräten verbunden."*
- Wenn die API `Generate` mit einem ungültigen Gerätenamen aufgerufen wird, wird eine Ausnahme gemeldet - *"HMI-Gerät 'Gerätename' nicht gefunden"*.
- Wenn die API `Generate` mit einem ungültigen PLC-Namen aufgerufen wird, wird eine Ausnahme gemeldet - *"PLC-Gerät 'PLC-Gerätename' nicht gefunden"*.
- Wenn die API `Generate` mit einem nicht unterstützten Gerätenamen aufgerufen wird, wird eine Ausnahme gemeldet - *"HMI-Gerät 'Gerätename' wird nicht unterstützt"*.
- Wenn die API `Generate` mit einem nicht unterstützten PLC-Namen aufgerufen wird, wird eine Ausnahme gemeldet - *"PLC-Gerät 'PLC-Gerätename' wird nicht unterstützt"*.
- Wenn die API `Generate` mit einer Liste der PLC-Namen aufgerufen wird, die den PLC-Namen nicht enthält, der bei der vorherigen SiVArc-Generierung verwendet wurde, berücksichtigt das System die zuvor eingefrorene und neu übergebene Liste der PLC-Gerätenamen für die Generierung.
- Wenn die API `Generate` mit einer Liste der HMI-Geräte/Panels aufgerufen wird und jedes HMI-Gerät/Panel mit einer PLC verbunden ist, wird die SiVArc-Generierung erfolgreich ausgeführt.

5.19 Unterstützung für Openness

5.19.1 Einleitung

Übersicht

-TIA Portal Test Suite über Openness API unterstützt eine Reihe von Funktionen für definierte Aufgaben, die Sie mit der TIA Portal Openness API außerhalb von TIA Portal aufrufen können.

Die folgenden Abschnitte enthalten eine Übersicht der typischen Programmierschritte. Sie erfahren, wie die einzelnen Codeabschnitte miteinander verknüpft sind und wie die jeweiligen Funktionen in ein komplettes Programm integriert werden können. Ferner erhalten Sie einen Überblick über diejenigen Teile des Code, die für jede Aufgabe anzupassen sind.

Verbinden mit TIA Portal

Die TIA Portal Test Suite ist ein optionales Package in TIA Portal, das die Effizienz des PLC-Programmierers durch zuverlässig hohe Qualität steigert. Um die Test Suite in der Openness-Anwendung nutzen zu können, muss das Test Suite Package installiert werden.

Verfügbare Namensräume für Test Suite-Objekte

Die folgenden Namensräume stehen für die Test Suite-Schnittstellen für die Funktionen Styleguide und Applikationstest zur Verfügung:

- `Siemens.Engineering.TestSuite`
- `Siemens.Engineering.TestSuite.StyleGuide`
- `Siemens.Engineering.TestSuite.ApplicationTest`
- `Siemens.Engineering.TestSuite.SystemTest`
- `Siemens.Engineering.TestSuite.StyleGuide.RuleSetExecutor`
- `Siemens.Engineering.TestSuite.ApplicationTest.TestCaseExecutor`
- `Siemens.Engineering.TestSuite.SystemTest.TestCaseExecutor`
- `Siemens.Engineering.TestSuite.TestResults`

Zugriff auf Test Suite als Dienst

Anwendung

Sie können auf die Test Suite und ihre Methoden in der Openness-Anwendung im TIA Portal-Zielformat mit dem folgenden Code zugreifen:

Programmcode

Für den Zugriff auf Test Suite als Dienst:

```
Using Siemens.Engineering;
Using Siemens.Engineering.TestSuite;
```

```
TestSuiteService testSuite = project.GetService<TestSuiteService>( ) ;
```

'project' ist hier eine Instanz des TIA Portal-Projekts, die in die openness-Anwendung geladen wurde.

Hinweis

Der obige Programmcode erzeugt einen Build-Fehler, wenn das Test Suite Package nicht installiert ist.

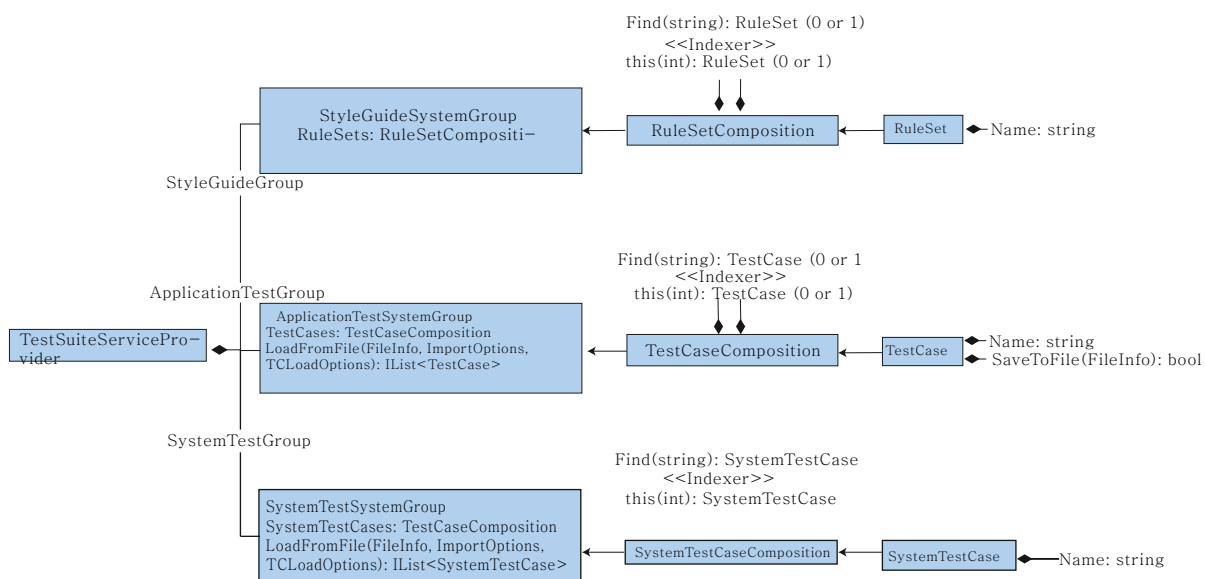
Funktionen

Im folgenden Abschnitt sind die Funktionen für definierte Aufgaben beschrieben, die Sie mit TIA Portal Openness außerhalb des TIA Portal aufrufen können.

Test Suite Openness-Objektmodell

Übersicht

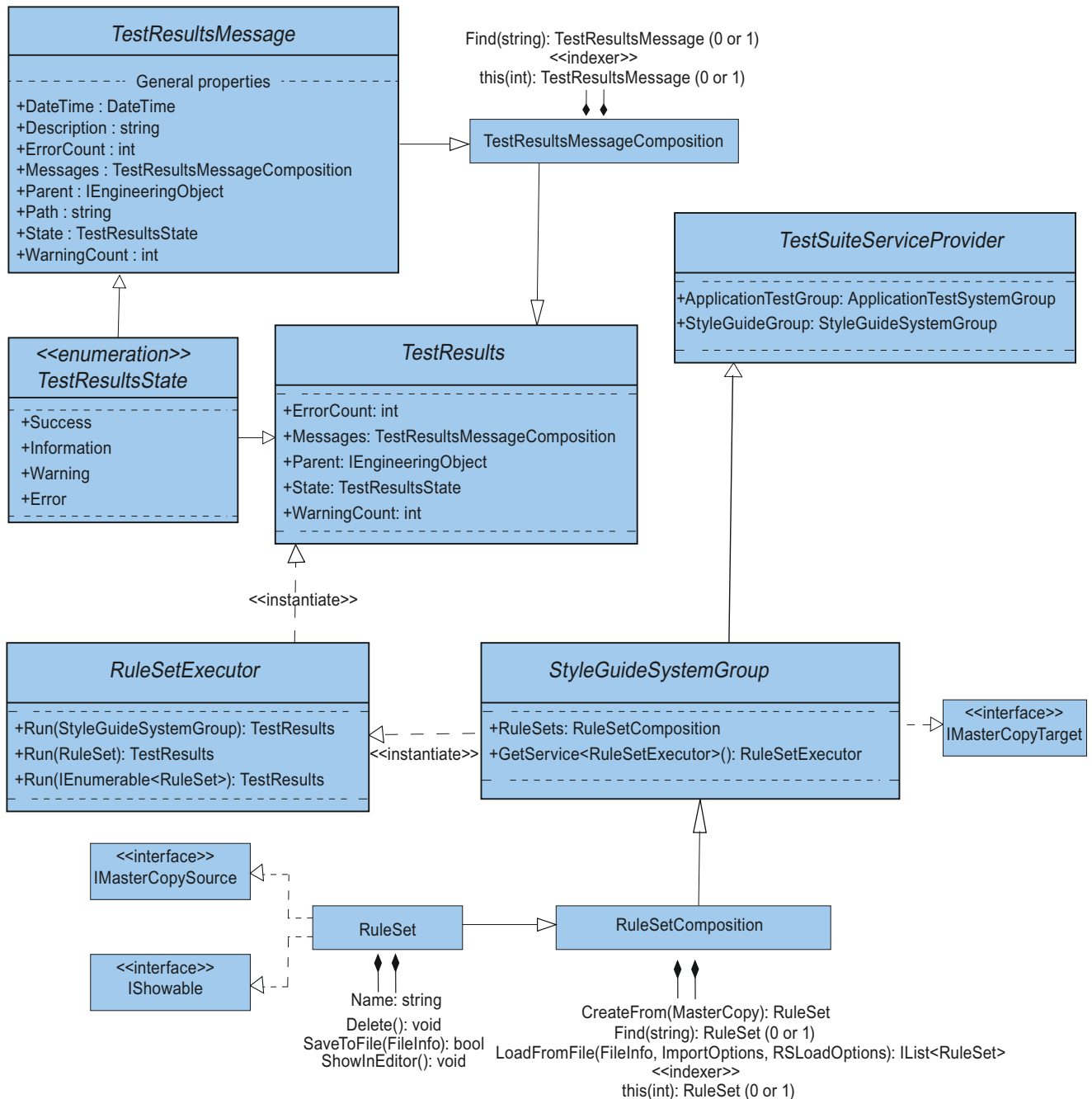
Das folgende Diagramm beschreibt die höchste Ebene des Objektmodells für Test Suite Openness:

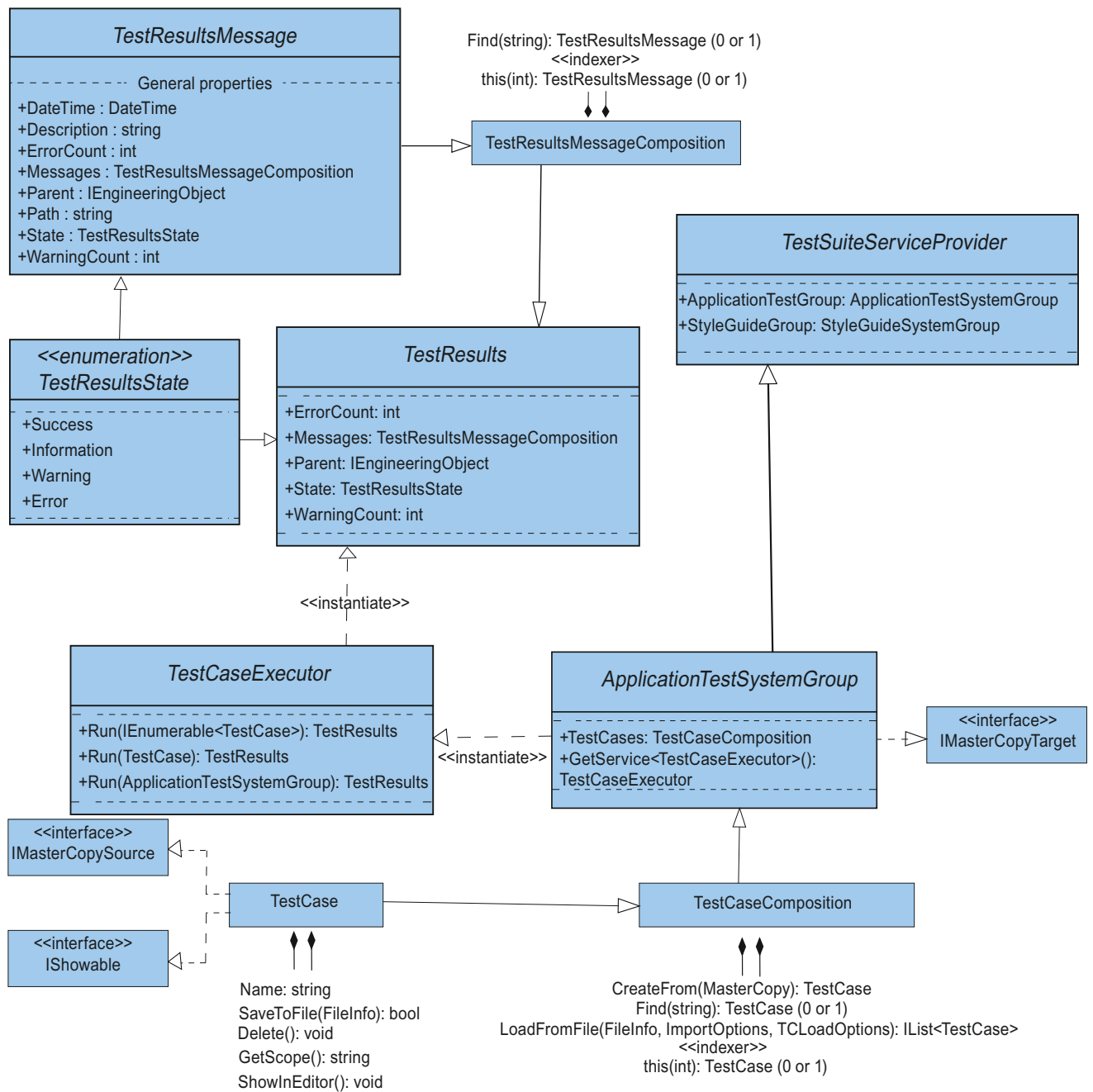


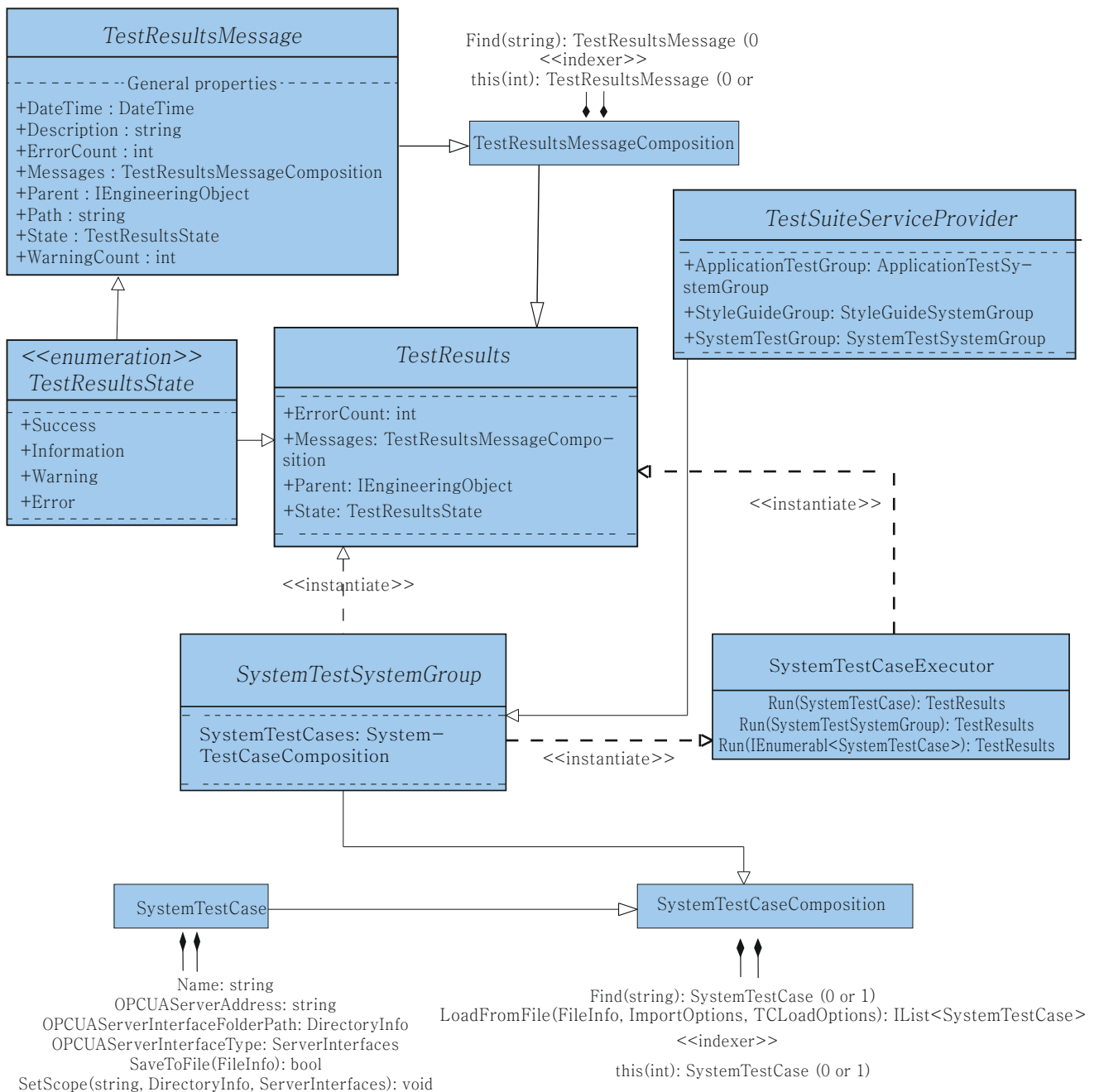
Klassendiagramme

Im Objektmodell von TIA Portal Openness sind alle Klassen, die nicht direkt instanziiert werden, als abstrakt definiert.

Das folgende Diagramm beschreibt die Objekte, die sich unter Testfallausführung und Kopiervorlagen befinden.







5.19.2 Funktionen für Styleguide

5.19.2.1 Zugriff auf Styleguide-Systemordner

Programmcode

Die folgenden Eigenschaften werden verwendet, um auf den Systemordner eines Styleguides als Systemgruppe zuzugreifen:

```
Using Siemens.Engineering.TestSuite.StyleGuide;  
  
StyleGuideSystemGroup scFolder = testSuite.StyleGuideGroup;
```

Referenz

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse für Styleguides:

Eigenschaftsname	Datentyp	Art des Zugriffs
StyleGuideGroup	TestSuite.StyleGuide.StyleGuideSystemGroup	Lesen

5.19.2.2 Zugriff auf die Regelsatz-Zusammensetzung

Programmcode

Mit der folgenden Eigenschaft werden alle Regelsätze im Projekt aufgerufen:

```
RuleSetComposition listRuleSets = testSuite.StyleGuideGroup.RuleSets;
```

Referenz

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse für Styleguides:

Eigenschaftsname	Datentyp	Art des Zugriffs
RuleSets	TestSuite.RuleSetComposition	Lesen

5.19.2.3 Zugriff auf Regelsatz

Beschreibung

Regelsätze in einem TIA Portal-Projekt können auf die folgenden zwei verschiedenen Arten in die Openness-Applikation geladen werden:

1. Indexierter Zugriff auf RuleSetComposition.
2. Verwendung von Find() method in RuleSetComposition.

Programmcode

Mit den folgenden Eigenschaften kann auf einen Regelsatz aus dem Projekt zugegriffen und ein spezieller Regelsatz aus dem Projekt geladen werden:

```
RuleSet rulesetDB = testSuite.StyleGuideGroup.RuleSets[1];  
RuleSet ruleSet1 = testSuite.StyleGuideGroup.RuleSets.Find("rulesetDB");
```

Referenz

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse für Styleguides:

Eigenschaftsname	Datentyp	Art des Zugriffs
RuleSet	RuleSetComposition	Lesen

Methoden:

```
public RuleSet Find(string Name)
```

5.19.2.4 Zugriff auf den Namen des Regelsatzes

Zugriff auf den Namen des Regelsatzes

Mit der folgenden Eigenschaft können Sie auf den Namen des Regelsatzes zugreifen:

```
string ruleSetName = testSuite.StyleGuideGroup.RuleSets[0].Name;
```

Umbenennen des ausgewählten Regelsatzes in einen gültigen Namen

Mit der unten genannten Eigenschaft können Sie den ausgewählten Regelsatz in einen gültigen Namen umbenennen.

```
public string Name { get; set; }
```

Hinweis

Mit der obigen Eigenschaft "Name" können Sie die folgenden möglichen Aufgaben durchführen:

- Den Namen des vorhandenen Regelsatzes lesen
- Den Namen des vorhandenen Regelsatzes ändern

Referenz

Die folgende Tabelle beschreibt die Eigenschaften der Klasse für Styleguides:

Eigenschaftsname	Datentyp	Art des Zugriffs
Name	String	Lesen/Schreiben

Programmcode

Sie können den folgenden Programmcode modifizieren, um einen Regelsatz in einen gültigen Namen umzubenennen:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.StyleGuide;

ProjectBase tiaProject = ... ;
TestSuiteService testSuite = tiaProject.GetService<TestSuiteService>();
RuleSetExecutor styleGuideService =
testSuite.StyleGuideGroup.GetService<RuleSetExecutor>();

//Read the name of a rule set
//Retrieve single rule set
RuleSet myRuleSet1 = testSuite.StyleGuideGroup.RuleSets[0];
String myRuleSet_name = myRuleSet1.Name;

//Modify the name of a rule set
myRuleSet1.Name = "myRuleSet_modified";
```

Hinweis

Wenn Sie einen falschen Namen eingeben, zeigt die Anwendung eine Ausnahme an (z. B. wenn Sie mit einem Whitespace beginnen oder der Name ein ungültiges Zeichen enthält).

5.19.2.5 Regelsatz einblenden**Öffnen des ausgewählten Regelsatzes in "WithUI"**

Mit der unten genannten Eigenschaft können Sie den ausgewählten Regelsatz im TIA Portal Editor im Modus "WithUI" öffnen.

```
public void ShowInEditor();
```

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um den Regelsatz im TIA Portal Editor im Modus "WithUI" zu öffnen:

```
using Siemens.Engineering;
using Siemens.Engineering.SW;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.StyleGuide;

//Retrives the first device available in the project
var device = tiaProject.Devices.First();

//Retrives "PLC_1" object from the above device
var deviceItem = device.DeviceItems.First(e => e.Classification ==
DeviceItemClassifications.CPU && e.Name == "PLC_1");

//Accessing rule set from same project
RuleSet myRuleSet = tiaProject.GetService<TestSuiteService>().StyleGuideGroup.RuleSets[1];

//Shows the selected item
myRuleSet.ShowInEditor();
```

5.19.2.6 Regelsatz löschen

Löschen des ausgewählten Regelsatzes in "WithUI/WithoutUI"

Mit der unten genannten Eigenschaft können Sie den ausgewählten Regelsatz aus dem TIA Portal-Projekt im Modus "WithUI/WithoutUI" löschen.

```
public void Delete();
```

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um den Regelsatz aus dem TIA Portal-Projekt im Modus "WithUI/WithoutUI" zu löschen:

```
using Siemens.Engineering;
using Siemens.Engineering.SW;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.StyleGuide;

//Retrives the first device available in the project
var device = tiaProject.Devices.First();

//Retrives "PLC_1" object from the above device
var deviceItem = device.DeviceItems.First(e => e.Classification ==
DeviceItemClassifications.CPU && e.Name == "PLC_1");

//Accessing rule set from same project
RuleSet myRuleSet = tiaProject.GetService<TestSuiteService>().StyleGuideGroup.RuleSets[1];

//Deletes the selected item
myRuleSet.Delete();
```

5.19.2.7 Den Geltungsbereich eines Regelsatzes modifizieren

Aktualisieren des Geltungsbereichs für einen bestehenden Regelsatz

Festlegen des Geltungsbereichs eines Regelsatzes

Sie können unten stehende API verwenden, um den Geltungsbereich eines vorhandenen Regelsatzes zu aktualisieren.

Referenz

Der Geltungsbereich eines Regelsatzes kann anhand der unten aufgeführten Methoden modifiziert werden:

- `SetScope()`
- `CopyScope()`

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public bool SetScope(UpdateOptions updateOptions,
IList<IEngineeringObject> step7Objects);
```


Die folgende Tabelle beschreibt die Eigenschaften der Klasse:

Name	Datentyp	Beschreibung
Parameter	<code>UpdateOptions.Override</code>	Überschreibt den bestehenden Geltungsbereich mit der neuen Liste der im selben Projekt verfügbaren Objekte
	<code>UpdateOptions.Add</code>	Hängt die neue Liste der im gleichen Projekt verfügbaren Objekte an den bestehenden Geltungsbereich an
	<code>ICollection<IEngineeringObject></code>	Die Liste der Objekte aus der unten stehenden Tabelle kann als Geltungsbereich für einen Regelsatz festgelegt werden

Zugriff auf Step7-Objekte, die dem Regelsatz als Geltungsbereich zugewiesen werden sollen

Step7-Objekte	Darstellung in der Openness-Anwendung (<code>IEngineeringObject</code>)	Beschreibung
Programmbausteine	<code>PlcBlockSystemGroup</code>	Alle verfügbaren Bausteine des Ordners "Programmbausteine" werden als Geltungsbereich festgelegt, mit Ausnahme jener des Ordners "Systembausteine".
PLC-Variablen	<code>PlcTagTableSystemGroup</code>	Alle verfügbaren Variablentabellen im Ordner "PLC-Variablen" werden als Geltungsbereich festgelegt.
PLC-Datentypen	<code>PlcTypeSystemGroup</code>	Alle verfügbaren UDTs des Ordners "PLC-Datentypen" werden als Geltungsbereich festgelegt, außer jene aus "Systemdatentypen".
Software-Units	<code>UnitGroup</code> <code>UnitGroup.Units[x]</code> <code>UnitGroup.Units[x].PlcBlockSystemGroup</code> <code>UnitGroup.Units[x].PlcTagTableSystemGroup</code> <code>UnitGroup.Units[x].PlcTypeSystemGroup</code>	Alle verfügbaren Units des Ordners "Software-Units" werden als Geltungsbereich festgelegt, darunter Programmbausteine, PLC-Variablen und PLC-Datentypen. Eine spezifische Auswahl ist im Units-Ordner ebenfalls möglich.
PLC/CPU	<code>DeviceItemClassifications.CPU</code> / <code>DeviceItems[0]</code>	Die gesamte "PLC" mit obigen Objekten wird als Geltungsbereich festgelegt
Projekt	<code>Project</code>	Das gesamte "Projekt" mit obigen Objekten wird als Geltungsbereich festgelegt

Step7-Objekte	Darstellung in der Openness-Anwendung (IEngineeringObject)	Beschreibung
Benutzergruppen	<code>PlcBlockSystemGroup.Groups[x]</code>	Eine bestimmte Benutzergruppe im Ordner "Programmbausteine" kann als Geltungsbereich festgelegt werden
	<code>PlcTagTableSystemGroup.Groups[x]</code>	Eine bestimmte Benutzergruppe im Ordner "PLC-Variablen" kann als Geltungsbereich festgelegt werden
	<code>PlcTypeSystemGroup.Groups[x]</code>	Eine bestimmte Benutzergruppe im Ordner "PLC-Datentypen" kann als Geltungsbereich festgelegt werden
	<code>Project.DeviceGroups[0]</code>	Eine Gruppe von PLCs kann unter dem Projekt als Geltungsbereich ausgewählt werden

Hinweis

Step7-Objekte, die nicht durch Styleguide unterstützt werden, werden während der Festlegung eines Geltungsbereichs und Ausführung eines Regelsatzes ignoriert.

Verwendete Methoden

Name der Methode	Beschreibung	Rückgabewert
<code>SetScope()</code>	Zur Modifikation des Geltungsbereichs eines bestehenden Regelsatzes	TRUE: Wenn die ausgewählten unterstützten Objekte als Geltungsbereich festgelegt sind
		FALSE: Wenn es nicht möglich ist, die ausgewählten unterstützten Objekte als Geltungsbereich festzulegen

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um die im ersten Geräteelement verfügbaren `IEngineeringObjects` abzurufen:

```
using Siemens.Engineering;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.StyleGuide;

// Retrives the first device available in the project
var device = tiaProject.Devices.First();

// To retrieve "PLC_1" object from the above device:
var deviceItem = device.DeviceItems.First(e => e.Classification
==DeviceItemClassifications.CPU && e.Name == "PLC_1");

// To retrieve the software content of the "PLC_1":
var plc = deviceItem.GetService<SoftwareContainer>().Software as PlcSoftware;

// To retrieve all the blocks available under "Program blocks" folder:
var plc1BlockGroup = plc.BlockGroup;

// To list blocks available under specific user group:
var plc1BlockGroupUserGroups = plc.BlockGroup.Groups[x];

// To provide all the tag tables available under "PLC tags" folder:
var plc1TagTableGroup = plc.TagTableGroup;

// To provide the PLC tag tables available under specific user group in "PLC Tags"
var plc1TagTableGroupUserGroups = plc.TagTableGroup.Groups[x];

// To provide all the UDTs available under "PLC data types" folder:
var plc1DataTypes = plc.TypeGroup;
var plc1DataTypes = plc.TypeGroup.Groups[x];

// To access all "Software units" available in "PLC_1":
var plc1SoftwareUnit = plc.GetService<PlcUnitProvider>().UnitGroup;

// To provide all the supported objects available in "Entire Project":
ProjectBase Project1 = tiaProject;

// To provide objects from group of devices in a project:
var GroupOfPLC1 = tiaProject.DeviceGroups[x];

// To access rule set from same project
RuleSet ruleSet1 = tiaProject.GetService<TestSuiteService>().StyleGuideGroup.RuleSets[1];

// update the scope of a ruleSet1:
// The below line of code appends the existing scope with new list of objects:
ruleSet1.SetScope(UpdateOptions.Add, new List<IEngineeringObject> { plc1BlockGroup,
plc1DataTypes, plc1TagTableGroup, plc1SoftwareUnit });

// The below code overwrites the existing scope with new objects:
ruleSet1.SetScope(UpdateOptions.Override, new List<IEngineeringObject>{GroupOfPLC1 });
```

Zuweisen des Geltungsbereichs von einem Regelsatz zu einem anderen Regelsatz

Sie können folgende API verwenden, um den Geltungsbereich von einem Regelsatz in einen anderen Regelsatz desselben Projekts zu kopieren.

Referenz

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public bool CopyScope(RuleSet targRuleSet);
```

Die folgende Tabelle beschreibt die Eigenschaften der Klasse:

Name	Datentyp	Beschreibung
targRuleSet	RuleSet	Zielregelsatz zur Aktualisierung des Geltungsbereichs

Verwendete Methoden

Name der Methode	Beschreibung	Rückgabewert
CopyScope()	Zur Zuweisung des Geltungsbereichs von einem Regelsatz zu einem anderen Regelsatz	TRUE: Wenn der Geltungsbereich erfolgreich kopiert wurde FALSE: Wenn der Geltungsbereich nicht erfolgreich kopiert wurde

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden:

```
// To access source and target rule sets from same project:
RuleSet srcRuleSet = tiaProject.GetService<TestSuiteService>().StyleGuideGroup.RuleSets[1];
RuleSet targRuleSet =
tiaProject.GetService<TestSuiteService>().StyleGuideGroup.RuleSets[10];

RuleSet srcRuleSet1 =
tiaProject.GetService<TestSuiteService>().StyleGuideGroup.Find("myRS");
RuleSet targRuleSet1 =
tiaProject.GetService<TestSuiteService>().StyleGuideGroup.Find("PLC");

// To assign scope from source to target rule set:
bool retVal = srcRuleSet.CopyScope(targRuleSet);
bool retVal1 = srcRuleSet1.CopyScope(targRuleSet1);
```

5.19.2.8 Ausführung von Regelsätzen

Die Ausführung eines Regelsatzes kann auf die folgenden drei Weisen in der Anwendung Openness erfolgen:

1. Bei einem bestimmten Regelsatz.
2. Bei einer Liste von Regelsätzen.
3. Alle verfügbaren Regelsätze im Styleguide-Ordner.

Referenz

Signatur

```
public RuleSetExecutor GetService<RuleSetExecutor>();
public TestResults Run(RuleSet ruleSet);
public TestResults Run(new IEnumerable<RuleSet>() {myRuleSet1,myStyleSet2,.. });
public TestResults Run(StyleGuideGroup allRuleSets);
```

Die folgenden Attribute werden von TestResults unterstützt:

Attributbezeichnung	Datentyp	Art des Zugriffs	Beschreibung
Messages	Siemens.Engineering.TestSuite.TestResultsMessgeComposition	Lesen	Zusammensetzung der Rückmeldung
Parent	IEngineeringObject	Lesen	Liefert das Parent-Element des Containers zurück
State	Siemens.Engineering.TestSuite.TestResultsState	Lesen	Endzustand in einer Testergebnis-Rückmeldung mit den möglichen Werten: Warnung, Fehler und Erfolg
WarningCount	Int	Lesen	Anzahl der Warnungen in einer Testergebnis-Rückmeldung
ErrorCount	Int	Lesen	Anzahl der Fehler in einer Testergebnis-Rückmeldung
Description	String	Lesen	Beschreibung oder Inhalt einer Testergebnis-Rückmeldung
DateTime	DateTime	Lesen	Datum und Uhrzeit einer Testergebnis-Rückmeldung
Path	String	Lesen	Pfad zu einer Testergebnis-Rückmeldung

Verwendete Methoden

Name der Methode	Beschreibung
Run ()	Zur Ausführung der ausgewählten Regelsätze

Hinweis

Format des Zeitstempels

Alle Zeitstempel sind in UTC angegeben. Wenn Sie sich die Ortszeit anzeigen lassen möchten, verwenden Sie `DateTime.ToLocalTime()`.

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.StyleGuide;

Project tiaProject = ... ;
TestSuiteService testSuite = tiaProject.GetService<TestSuiteService>();
RuleSetExecutor styleGuideService =
testSuite.StyleGuideGroup.GetService<RuleSetExecutor>();

//To execute specific rule set:
//Retrieve single rule set:
//Rule set at 0th index is available in the project:
RuleSet myRuleSet1 = testSuite.StyleGuideGroup.RuleSets[0];

//RuleSet_10 is available in the project:
RuleSet myStyleSet2 = testSuite.StyleGuideGroup.RuleSets.Find("RuleSet_10");

//Execute above retrieved rule sets:
TestResults ruleSetResults1 = styleGuideService.Run(myRuleSet1 );
TestResults ruleSetResults2 = styleGuideService.Run(myStyleSet2 );

//To execute list of rule sets:
//Retrieve list of rule sets to execute:
IEnumerable<RuleSet> ruleSetList = new List<RuleSet>() { myRuleSet1,myStyleSet2 };

//Execute above retrieved list of rule sets:
TestResults ruleSetsResults3 = styleGuideService.Run(ruleSetList );

//To execute all rule sets in StyleGuideGroup:
//Retrieve list of rule sets to execute:
StyleGuideSystemGroup styleGuideFolder = testSuite.StyleGuideGroup;

//Execute all available rule sets in Styleguide folder:
TestResults ruleSetsResults4 = styleGuideService.Run(styleGuideFolder);
```

Der Anwender kann auf die obigen Testergebnisse zugreifen und mithilfe des folgenden Codes durch die TestResults-Meldungen iterieren:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.StyleGuide;

...

//Retrieve test results of single rule set execution
WriteTestResults(ruleSetResults1);

//Retrieve test results of list of rule sets execution
WriteTestResults(ruleSetsResults3);

//Retrieve test results of rule sets execution available
in "StyleGuideGroup"
WriteTestResults(ruleSetsResults4);

...

private void WriteTestResults (TestResults result)
{
    Console.WriteLine("State:" + result.State);
    Console.WriteLine("Warning Count:" + result.WarningCount);
    Console.WriteLine("Error Count:" + result.ErrorCount);

    RecursivelyWriteMessages(result.Messages);
}

private void RecursivelyWriteMessages(TestResultsMessageComposition
messages,string indent = "")
{
    indent += "\t";
    foreach (TestResultsMessage message in messages)
    {
        Console.WriteLine(indent + "Path: " + message.Path);
        Console.WriteLine(indent + "DateTime: " + message.DateTime);
        Console.WriteLine(indent + "State: " + message.State);
        Console.WriteLine(indent + "Description: " + message.Description);
        Console.WriteLine(indent + "Warning Count: " + message.WarningCount);
        Console.WriteLine(indent + "Error Count: " + message.ErrorCount);

        RecursivelyWriteMessages(message.Messages, indent);
    }
}
```

5.19.2.9 Import und Export von Regelsätzen

Verwendete Methoden

Zum Importieren und Exportieren von Regelsätzen werden die folgenden Methoden verwendet:

Name der Methode	Beschreibung
SaveToFile()	Für den Export des Regelsatzes in eine externe Quelldatei
LoadFromFile()	Für den Import des Regelsatzes / der Regelsätze aus einer externen Quelldatei

Speichern eines Regelsatzes in einer externen Datei

Anforderung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.

Applikation

Mit Hilfe der Methode `SaveToFile()` exportieren Sie einen bestimmten Regelsatz in einen ausgewählten Pfad.

Signatur

```
public void SaveToFile(FileInfo path);
```

In der folgenden Tabelle sind die erforderlichen Parameter für die Methode aufgeführt:

Name des Parameters	Beschreibung
FileInfo	Vollständiger Pfad des zu speichernden Regelsatzes

Programmcode

`styleChecks` und `styleChecks1` sind hier in "MyProject" verfügbare Regelsätze.

```
RuleSetComposition styleChecks = testSuite.StyleGuideGroup.RuleSets;  
RuleSetComposition styleChecks1 = testSuite.StyleGuideGroup.RuleSets;  
styleChecks.SaveToFile(exportFile);  
styleChecks1.SaveToFile(new FileInfo(@"D:\Temp\TSRuleSets1.xml"));
```

Laden von Regelsätzen von einer externen Textdatei in ein Projekt

Anforderung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.

Applikation

Um einen oder mehrere Regelsätze von einer ausgewählten Datei zu importieren verwenden Sie die Methode `LoadFromFile()` auf `RuleSetComposition`.

Signatur

```
public IEnumerable<RuleSet> LoadFromFile (FileInfo path,
ImportOptions loadOptions, RSLoadOptions rsLoadOptions);
```

In der folgenden Tabelle sind die erforderlichen Parameter für die Methode aufgeführt:

Name des Parameters	Beschreibung
FileInfo	Vollständiger Pfad des zu speichernden Regelsatzes
ImportOptions.None	Eine Ausnahme wird ausgelöst, wenn der Regelsatz im Projekt unter demselben Namen bereits existiert/wenn die Datei mehrere Regelsätze mit demselben Namen enthält.
ImportOptions.Override	Überschreibt einen bestehenden Regelsatz im Projekt mit demselben Namen
RSLoadOptions.IgnorePropertyErrors	Importiert den Regelsatz / die Regelsätze mit ungültigen Eigenschaftswerten unter Author/Version/Comment
RSLoadOptions.None	Importiert den Regelsatz / die Regelsätze nur, sofern in der Zieldatei keine Warnungen/Fehler vorliegen
RSLoadOptions.IgnoreMissingAttributes	Importiert den Regelsatz / die Regelsätze mit fehlenden Regelattributen durch Einsetzen von Standardwerten
RSLoadOptions.SkipInvalidObjects	Importiert den Regelsatz / die Regelsätze durch Überspringen der Regeln mit ungültigen Attributen

Beispiele:

```
// Throw Exception if the rule set with the same name exists already
// Import the rule set with a invalid properties.

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSRuleSets1.xml");
styleChecks.LoadFromFile(LoadFile, ImportOptions.None,
RSLoadOptions.IgnorePropertyErrors);

// Import rule set with the same name exists already
// Throw exception and abort the import of rule set when there are errors/warnings

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSRuleSets1.xml");
styleChecks.LoadFromFile(LoadFile, ImportOptions.None, RSLoadOptions.None);

// Import rule set with the same name exists already

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSRuleSets1.xml");
styleChecks.LoadFromFile(LoadFile, ImportOptions.Override, RSLoadOptions...);
```

```
// Overwrite rule set with the same name exists already in project

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSRuleSets1.xml");
styleChecks.LoadFromFile(LoadFile, ImportOptions.Override, RSLoadOptions...);

// Import rule sets with the default values for the rules with missing attributes

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSRuleSets1.xml");
styleChecks.LoadFromFile(LoadFile, ImportOptions...,
RSLoadOptions.IgnoreMissingAttributes);

// Import rule sets and skip the import for the rules with invalid attributes

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSRuleSets1.xml");
styleChecks.LoadFromFile(LoadFile, ImportOptions..., RSLoadOptions.SkipInvalidObjects);

// Import the rule set by replacing missing rule attributes with default values and skipping
rules with invalid attributes

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSRuleSets1.xml");
styleChecks.LoadFromFile(LoadFile, ImportOptions..., RSLoadOptions.IgnoreMissingAttributes
| RSLoadOptions.SkipInvalidObjects);

// Imports the rule set(s) with errors in rule set property values, replaces the missing
rule attribute values with the default ones and skips the rules with invalid attributes

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSRuleSets1.xml");
styleChecks.LoadFromFile(LoadFile, ImportOptions..., RSLoadOptions.IgnoreMissingAttributes
| RSLoadOptions.SkipInvalidObjects | RSLoadOptions.IgnorePropertyErrors);
```

5.19.2.10 Arbeiten mit Bibliotheken

Regelsatz in Bibliotheken kopieren

Regelsätze aus dem Projekt können zur Speicherung in die Bibliotheken kopiert werden, damit sie wiederverwendet und abgeändert werden können. Sie müssen die Regelsätze für eine externe Änderung nicht explizit exportieren/importieren.

Wird die Projektbibliothek im Schreib-Lese-Modus geöffnet, können Sie am Zielort eine MasterCopy von einer IMasterCopySource erstellen.

Referenz

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public MasterCopy Create (IMasterCopySource sourceObject);
```

Verwendete Methoden

Name der Methode	Beschreibung
Create()	Zum Kopieren eines Testfalls von einem Projekt in eine Projekt- bzw. globale Bibliothek

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um einen Regelsatz von einem Projekt in die Bibliothek zu kopieren:

```
using Siemens.Engineering;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.StyleGuide;
using Siemens.Engineering.Library.MasterCopies;
...

// Retrieving rule set to copy to libraries:
RuleSet myRuleSet = testSuite.StyleGuideGroup.RuleSets.Find("CalcRuleSet");

// For copying a rule set to project library master copy system folder:
MasterCopy masterCopyInSystemFolder =
tiaProject.ProjectLibrary.MasterCopyFolder.MasterCopies.Create(myRuleSet );

// For copying a rule set to project library master copy user folder:
MasterCopy masterCopyinUserFolder =
tiaProject.ProjectLibrary.MasterCopyFolder.Folders[0].MasterCopies.Create(myRuleSet );
tiaPortal.GlobalLibraries.Open(new FileInfo(@"D:\Documents\.."), OpenMode.ReadWrite);

// For copying a rule set to global library master copy system folder:
MasterCopy masterCopyInSystemFolder =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.MasterCopies.Create(myRuleSet );

// For copying a rule set to global library master copy user folder:
MasterCopy masterCopyinUserFolder =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.Folders[0].MasterCopies.Create(myRuleSet );
```

Masterkopien aus Projekten kopieren

Die API-Schnittstelle von TIA Portal Openness unterstützt das Kopieren von Masterkopien in Projekte mit Hilfe der Aktion `CreateFrom`. Die Aktion erstellt ein neues Objekt auf Grundlage der Masterkopie und platziert es in der Struktur, in der die Aktion aufgerufen wurde. Die Aktion versucht, den oder die neuen Regelsätze mit demselben Namen wie die Quell-Masterkopie zu erstellen. Wenn ein solcher Name in `RuleSetComposition` bereits besteht, gibt das System dem neuen Regelsatz einen neuen Namen. Dann wird der neue Regelsatz kopiert zurückgegeben.

Referenz

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public RuleSet CreateFrom (MasterCopy masterCopy);
```

Verwendete Methoden

Name der Methode	Beschreibung
<code>CreateFrom()</code>	Zum Kopieren von Masterkopien aus projektspezifischen oder globalen Bibliotheken in das Projekt

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um Masterkopien von einem Projekt in die Bibliothek zu kopieren:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.StyleGuide;
using Siemens.Engineering.Library.Mastercopies;

...

// For accessing master copy object in global library to copy to project:
var masterCopyObjectinGL = tiaPortal.GlobalLibraries[0].MasterCopyFolder.MasterCopies[0];
var masterCopyObjectinUserFolderinGL =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.Folders[0].MasterCopies[1];

// To copy rule set from global library system folder to project:
RuleSet RuleSet1 =testSuite.StyleGuideGroup.RuleSets.CreateFrom(masterCopyObjectinGL);

// To copy rule set from global library user folder to project:
RuleSet RuleSet2
=testSuite.StyleGuideGroup.RuleSets.CreateFrom(masterCopyObjectinUserFolderinGL);

// For accessing master copy object in project library to copy to project:
var masterCopyObjectinPL =tiaProject.ProjectLibrary.MasterCopyFolder.MasterCopies[0];
var masterCopyObjectinUserFolderinPL
=tiaProject.ProjectLibrary.MasterCopyFolder.Folders[0].MasterCopies[0];

// To copy rule set from project library system folder to project:
RuleSet RuleSet3 =testSuite.StyleGuideGroup.RuleSets.CreateFrom(masterCopyObjectinPL);

// To copy rule set from project library user folder to project:
RuleSet RuleSet4
=testSuite.StyleGuideGroup.RuleSets.CreateFrom(masterCopyObjectinUserFolderinPL);
```

Masterkopien (des Typs Regelsatz) innerhalb oder zwischen Bibliotheken kopieren

Informationen zum Kopieren von Masterkopien (vom Typ TestCase) innerhalb von und zwischen Bibliotheken finden Sie in **TIA Portal Openness API** unter **Funktionen für Bibliotheken** im Punkt **Masterkopien kopieren** im Informationssystem.

5.19.3 Funktionen für den Anwendungstest

5.19.3.1 Zugriff auf den Systemorder für den Anwendungstest

Programmcode

Mit den folgenden Eigenschaften können Sie auf den Systemordner einer Test Suite als Systemgruppe zugreifen:

```
using Siemens.Engineering;  
using Siemens.Engineering.TestSuite;  
using Siemens.Engineering.TestSuite.ApplicationTest;
```

```
TestSuiteService testSuite = project.GetService<TestSuiteService>( ) ;  
ApplicationTestSystemGroup atfFolder = testSuite.ApplicationTestGroup;
```

'project' ist hier eine Instanz des TIA Portal-Projekts, die in die openness-Anwendung geladen wurde.

Referenz

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse für den Applikationstest:

Eigenschaftsname	Datentyp	Art des Zugriffs
ApplicationTestGroup	TestSuite.ApplicationTest.ApplicationTestSystemGroup	Lesen

5.19.3.2 Zugriff auf TestCaseComposition

Programmcode

Mit der folgenden Eigenschaft werden alle verfügbaren Testfälle im Projekt aufgerufen:

```
TestCaseComposition listTestCases =  
testSuite.ApplicationTestGroup.TestCases;
```

Referenz

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse für den Applikationstest:

Eigenschaftsname	Datentyp	Art des Zugriffs
TestCases	TestSuite.TestCaseComposition	Lesen

5.19.3.3 Zugriff auf den Testfall

Beschreibung

Testfälle in einem TIA Portal-Projekt können auf die folgenden zwei verschiedenen Arten in die Openness-Applikation geladen werden:

1. Indexierter Zugriff auf TestCaseComposition.
2. Verwendung von Find() method in TestCaseComposition.

Programmcode

Mit den folgenden Eigenschaften können Sie auf einen einzelnen Testfall im Projekt zugreifen und einen bestimmten Testfall aus dem Projekt laden:

```
TestCase motorFCTestcase = testSuite.ApplicationTestGroup.TestCases[1];  
TestCase testCase = testSuite.ApplicationTestGroup.TestCases.Find("TCMot");
```

Referenz

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse für den Applikationstest:

Eigenschaftsname	Datentyp	Art des Zugriffs
TestCases	TestCaseComposition	Lesen

Methoden:

```
public TestCase Find(string Name)
```

5.19.3.4 Zugriff auf den Namen des Testfalls

Zugriff auf den Namen des Testfalls

Mit der folgenden Eigenschaft können Sie auf den Namen des Testfalls zugreifen:

```
string testCaseName = testSuite.ApplicationTestGroup.TestCases[0].Name;
```

Umbenennen des ausgewählten Testfalls in einen gültigen Namen

Mit der unten genannten Eigenschaft können Sie den ausgewählten Testfall in einen gültigen Namen umbenennen.

```
public string Name { get; set; }
```

Hinweis

Mit der obigen Eigenschaft "Name" können Sie die folgenden möglichen Aufgaben durchführen:

- Den Namen des vorhandenen Testfalls lesen
- Den Namen des vorhandenen Testfalls ändern

Referenz

Die folgende Tabelle beschreibt die Eigenschaften der Klasse für Applikationstests:

Eigenschaftsname	Datentyp	Art des Zugriffs
Name	String	Lesen/Schreiben

Programmcode

Sie können den folgenden Programmcode modifizieren, um einen Testfall in einen gültigen Namen umzubenennen:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

ProjectBase tiaProject = ... ; TestSuiteService testSuite =
tiaProject.GetService<TestSuiteService>();
TestCaseExecutor applicationTestService =
testSuite.ApplicationTestGroup.GetService<TestCaseExecutor>();

//Read the name of a test case
//Retrieve single test case
TestCase myTestCase1 = testSuite.ApplicationTestGroup.TestCases[0];
String myTestCase_name = myTestCase1.Name;

//Modify the name of a test case
myTestCase1.Name = "myTestCase_modified";
```

Hinweis

Wenn Sie einen falschen Namen eingeben, zeigt die Anwendung eine Exception an (z. B. wenn Sie mit einem Whitespace beginnen oder der Name ein ungültiges Zeichen enthält).

5.19.3.5 Zugriff auf den Scope des Testfalls**Zugriff auf den Scope eines vorhandenen Testfalls**

Die folgende Methode dient zum Zugriff auf den Scope eines vorhandenen Testfalls:

GetScope()

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public plcSoftware GetScope();
```

Die folgende Tabelle beschreibt die Eigenschaft der Klasse:

Name	Datentyp	Beschreibung
Parameter	String	Name des als Scope für den Testfall ausgewählten PLC

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um auf den Scope des vorhandenen Testfalls zuzugreifen:

```
using Siemens.Engineering;
using Siemens.Engineering.SW;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

//Retrives the first device available in the project
var device = tiaProject.Devices.First();

//Retrives "PLC_1" object from the above device
var deviceItem = device.DeviceItems.First(e => e.Classification ==
DeviceItemClassifications.CPU && e.Name == "PLC_1");

//Accessing test case from same project
TestCase myTestCase =
tiaProject.GetService<TestSuiteService>().ApplicationTestGroup.TestCases[1];

//Access the scope of myTestCase
PlcSoftware twoHandMonitor = myTestCase.GetScope();
```

5.19.3.6 Testfall einblenden

Ausgewählten Testfall im Modus "WithUI" öffnen

Die nachstehend genannte Eigenschaft dient dazu, den ausgewählten Testfall im TIA Portal-Editor im Modus "WithUI" zu öffnen.

```
public void ShowInEditor();
```

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um den Testfall im TIA Portal-Editor im Modus "WithUI" zu öffnen:

```
using Siemens.Engineering;
using Siemens.Engineering.SW;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

//Retrives the first device available in the project
var device = tiaProject.Devices.First();

//Retrives "PLC_1" object from the above device
var deviceItem = device.DeviceItems.First(e => e.Classification ==
DeviceItemClassifications.CPU && e.Name == "PLC_1");

//Accessing test case from same project
TestCase myTestCase =
tiaProject.GetService<TestSuiteService>().ApplicationTestGroup.TestCases[1];

//Shows the selected item
myTestCase.ShowInEditor();
```

5.19.3.7 Testfall löschen

Ausgewählten Testfall im Modus "withUI/WithoutUI" löschen

Die nachstehend genannte Eigenschaft dient dazu, den ausgewählten Testfall im Modus "withUI/WithoutUI" aus dem TIA Portal-Projekt zu löschen.

```
public void Delete();
```

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um den Testfall im Modus "withUI/WithoutUI" aus dem TIA Portal-Projekt zu löschen:

```
using Siemens.Engineering;
using Siemens.Engineering.SW;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

//Retrives the first device available in the project
var device = tiaProject.Devices.First();

//Retrives "PLC_1" object from the above device
var deviceItem = device.DeviceItems.First(e => e.Classification ==
DeviceItemClassifications.CPU && e.Name == "PLC_1");

//Accessing test case from same project
TestCase myTestCase =
tiaProject.GetService<TestSuiteService>().ApplicationTestGroup.TestCases[1];

//Deletes the selected item
myTestCase.Delete();
```

5.19.3.8 Den Geltungsbereich eines Testfalls modifizieren

Den Geltungsbereich für einen vorhandenen Testfall aktualisieren

Den Geltungsbereichs eines Testfalls festlegen

Sie können unten stehende API verwenden, um den Geltungsbereich eines vorhandenen Testfalls zu aktualisieren.

Referenz

Der Geltungsbereichs eines Testfalls kann wie folgt modifiziert werden:

- `SetScope()`

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public void SetScope(plcSoftware PLC);
public void SetScope(plcSoftware PLC, string instanceName, ExecutionMode tcExecution);
```

Die folgende Tabelle beschreibt die Eigenschaften der Klasse:

Parameter	Datentyp	Beschreibung
plcSoftware	plcSoftware	CPU-Typen sind gültige Objekte, die mit PLCSIM Advanced kompatibel sind
Instanzname	string	Name der PLC Advanced-Instanz
tcExecutionMode	Ausführungsmodus	Auswahl des Testfall-Ausführungsmodus mit ExternallyManagedPLCSIMInstance / SystemManagedPLCSIMInstance

Hinweis

Es ist nicht möglich, den Geltungsbereich für Software Controller mit SetScope() API festzulegen.

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden:

```
using Siemens.Engineering;
using Siemens.Engineering.SW;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

//Retrives the first device available in the project
var device = tiaProject.Devices.First();

//To retrieve "PLC_1" object from the above device:
var deviceItem = device.DeviceItems.First(e => e.Classification
==DeviceItemClassifications.CPU && e.Name == "PLC_1");

//To retrieve the software content of the "PLC_1":
var twoHandMonitor = deviceItem.GetService<SoftwareContainer>().Software
as PlcSoftware;

//To access test case from same project
TestCase myTestCase =
tiaProject.GetService<TestSuiteService>().ApplicationTestGroup.TestCases[1
];

//Updates the scope of myTestCase:
myTestCase.SetScope(twoHandMonitor);
```

```

using Siemens.Engineering;
using Siemens.Engineering.SW;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

//Retrives the first device available in the project
var device = tiaProject.Devices.First();

//Creates a new TIA portal instance
TiaPortal tiaPortal = new TiaPortal(TiaPortalMode.WithUserInterface);

//Assigns the path of the project to the project path variable
FileInfo projectPath = new FileInfo(project_path);

//Opens the TIA project AND Test Suite
Project tiaProject = tiaPortal.Projects.Open(projectPath);
TestSuiteService testSuite = tiaProject.GetService<TestSuiteService>();

//To retrieve "PLC_1" object from the above device:
var deviceItem = device.DeviceItems.First(e => e.Classification ==
DeviceItemClassifications.CPU && e.Name == "PLC_1");
//To retrieve the software content of the "PLC_1":
var plc = deviceItem.GetService<SoftwareContainer>().Software as
PlcSoftware;

//Specifies the instance name
string instanceName = "SetScope";

//To access test case from same project
TestCase myTestcase = testSuite.ApplicationTestGroup.TestCases[1];

//Updates the scope of ExternallyManagedPLCSIMInstance:
myTestcase.SetScope(plc, instanceName,
ExecutionMode.ExternallyManagedPLCSIMInstance);

```

5.19.3.9 Ausführung von Testfällen

Die Ausführung eines Testfalls kann auf die folgenden drei Weisen in der Anwendung Openness erfolgen:

1. Bei einem bestimmten Testfall.
2. Bei einer Liste von Testfällen.
3. Bei allen verfügbaren Testfällen im Applikationstest-Ordner.

Referenz

Signatur

```
public TestCaseExecutor GetService<TestCaseExecutor>();
public TestResults Run(TestCase myTestCase);
public TestResults Run(new IEnumerable<TestCase>() {myTestCase1,myTestCase2,.. });
public TestResults Run(ApplicationTestGroup allTestCases);
```

Die folgenden Attribute werden von TestResults unterstützt:

Attributbezeichnung	Datentyp	Art des Zugriffs	Beschreibung
Messages	Siemens.Engineering.TestSuite.TestResultsMessageComposition	Lesen	Zusammensetzung der Rückmeldung
Parent	IEngineeringObject	Lesen	Liefert das Parent-Element des Containers zurück
State	Siemens.Engineering.TestSuite.TestResultsState	Lesen	Endzustand in einer Testergebnis-Rückmeldung mit den möglichen Werten: Warnung, Fehler und Erfolg
WarningCount	Int	Lesen	Anzahl der Warnungen in einer Testergebnis-Rückmeldung
ErrorCount	Int	Lesen	Anzahl der Fehler in einer Testergebnis-Rückmeldung
Beschreibung	String	Lesen	Beschreibung oder Inhalt einer Testergebnis-Rückmeldung
DateTime	DateTime	Lesen	Datum und Uhrzeit einer Testergebnis-Rückmeldung
Path	String	Lesen	Pfad zu einer Testergebnis-Rückmeldung

Verwendete Methoden

Name der Methode	Beschreibung
Run ()	Zur Ausführung der ausgewählten Testfälle

Hinweis

Format des Zeitstempels

Alle Zeitstempel sind in UTC angegeben. Wenn Sie sich die Ortszeit anzeigen lassen möchten, verwenden Sie `DateTime.ToLocalTime()`.

Hinweis

Sie können die Testfälle sowohl im Ausführungsmodus ExternallyManagedPLCSIMInstance oder SystemManagedPLCSIMInstance ausführen.

Weitere Informationen zu ExternallyManagedPLCSIMInstance finden Sie unter Ausführung von Testfällen mit ExternallyManagedPLCSIMInstance

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

ProjectBase tiaProject = ... ;
TestSuiteService testSuite = tiaProject.GetService<TestSuiteService>();
TestCaseExecutor applicationTestService =
testSuite.ApplicationTestGroup.GetService<TestCaseExecutor>();

//To execute specific test case:
//Retrieve single test case:
//Test case at 0th index is available in the project:
TestCase myTestCase1 = testSuite.ApplicationTestGroup.TestCases[0];

//TestCase_10 is available in the project:
TestCase myTestCase2 =
testSuite.ApplicationTestGroup.TestCases.Find("TestCase_10");

//Execute above retrieved test cases:
TestResults TestCaseResults1 = applicationTestService.Run(myTestCase1 );
TestResults TestCaseResults2 = applicationTestService.Run(myTestCase2 );

//To execute list of test cases:
//Retrieve list of test cases to execute:
IEnumerable<TestCase> TestCaseList = new List<TestCase>()
{ myTestCase1,myTestCase2 };

//Execute above retrieved list of test cases:
TestResults TestCasesResults3 = applicationTestService.Run(TestCaseList );

//To execute all test cases in ApplicationTestGroup:
//Retrieve list of test cases to execute:
ApplicationTestSystemGroup applicationTestFolder =
testSuite.ApplicationTestGroup;

//Execute all available test cases in ApplicationTestFolder:
TestResults TestCasesResults4 =
applicationTestService.Run(applicationTestFolder);
```

5.19 Unterstützung für Openness

Der Anwender kann auf die obigen Testergebnisse zugreifen und mithilfe des folgenden Codes durch die TestResults-Meldungen iterieren:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

...

//Retrieve test results of single test case execution
WriteTestResults(TestCaseResults1);

//Retrieve test results of list of test cases execution
WriteTestResults(TestCasesResults3);

//Retrieve test results of test cases execution available in "ApplicationTestGroup"
WriteTestResults(TestCasesResults4);

...

private void WriteTestResults (TestResults result)
{
    Console.WriteLine("State:" + result.State);
    Console.WriteLine("Warning Count:" + result.WarningCount);
    Console.WriteLine("Error Count:" + result.ErrorCount);

    RecursivelyWriteMessages(result.Messages);
}
private void RecursivelyWriteMessages (TestResultsMessageComposition messages, string indent
= "")
{
    indent += "\t";
    foreach (TestResultsMessage message in messages)
    {
        Console.WriteLine(indent + "Path: " + message.Path);
        Console.WriteLine(indent + "DateTime: " + message.DateTime);
        Console.WriteLine(indent + "State: " + message.State);
        Console.WriteLine(indent + "Description: " + message.Description);
        Console.WriteLine(indent + "Warning Count: " + message.WarningCount);
        Console.WriteLine(indent + "Error Count: " + message.ErrorCount);

        RecursivelyWriteMessages(message.Messages, indent);
    }
}
```

Sie können den nachstehenden Beispielcode kopieren und ändern, um Testfälle im "ExternallyManagedPLCSIMInstance-Modus" auszuführen.

5.19 Unterstützung für Openness

```

using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;
using Siemens.Simatic.Simulation.Runtime;

string m_PlcInstanceName;
IInstance m_PlcSimInstance;

//To create PLCSIM Advanced instance and power on

m_PlcSimInstance = SimulationRuntimeManager.RegisterInstance(ECPUType.CPU1500_Unspecified,
m_PlcInstanceName);
m_PlcSimInstance.PowerOn(60000);

ProjectBase tiaProject = ... ;

//Download the selected device item (Project PLC to PLCSIM instance created)

//Refer the chapter TIA Portal Openness API->Functions for accessing the data of a PLC
device->Functions for downloading data to PLC device ->Downloading to PLC devices

DeviceItem deviceItem = ...;
DownloadProvider downloadProvider = deviceItem.GetService<DownloadProvider>();
    if (downloadProvider != null)
    {
        ...
    }

//To retrieve the software content of the selected PLC to set the scope of test cases to run

var twoHandMonitor = deviceItem.GetService<SoftwareContainer>().Software as PlcSoftware;
TestSuiteService testSuite = tiaProject.GetService<TestSuiteService>();
TestCaseExecutor applicationTestService =
testSuite.ApplicationTestGroup.GetService<TestCaseExecutor>();

//To run list of test cases:

//Test case at 0th index is available in the project:

TestCase myTestCase1 = testSuite.ApplicationTestGroup.TestCases[0];
myTestCase1.SetScope(twoHandMonitor,m_PlcInstanceName,ExecutionMode.ExternallyManagedPLCSIMInstance);

//TestCase_10 is available in the project:

TestCase myTestCase2 = testSuite.ApplicationTestGroup.TestCases.Find("TestCase_10");
myTestCase2.SetScope(twoHandMonitor,m_PlcInstanceName,ExecutionMode.ExternallyManagedPLCSIMInstance);

//Retrieve list of test cases to execute:

IEnumerable<TestCase> TestCaseList = new List<TestCase>() { myTestCase1,myTestCase2 };

```

```
//Execute above retrieved list of test cases, Connects to the existing PLCSIM Advanced instance and runs the test cases selected
```

```
TestResults TestCasesResults = applicationTestService.Run(TestCaseList );
```

Arbeiten mit PLCSIM Advanced V6.0

Arbeiten mit dem TIA Portal im Legacy-Modus

Sie können das folgende Programmcodebeispiel modifizieren und verwenden:

```
//Initialize TIA Portal instance
var tiaPortalUi = new TiaPortal(TiaPortalMode.WithUserInterface);

//Open the project
var project = tiaPortalUi.Projects.Open(new FileInfo(projectPath));

//Enable legacy mode for the PLC

//For R/H type handling
var onlineProvider = device.GetService<RHOnlineProvider>();
var configuration = onlineProvider.Configuration;
if(!configuration.EnableLegacyCommunication) configuration.EnableLegacyCommunication = true;

//For any other CPU type
var onlineProvider = deviceItem.GetService<OnlineProvider>();
var configuration = onlineProvider.Configuration;
if(!configuration.EnableLegacyCommunication) configuration.EnableLegacyCommunication = true;

//Execute test case
var testSuiteService = project.GetService<TestSuiteService>();
var testCaseExecutor = testSuiteService.ApplicationTestGroup.GetService<TestCaseExecutor>();
testCaseExecutor.Run(...);
```

Arbeiten mit dem TIA Portal im sicheren Modus

Sie können das folgende Programmcodebeispiel modifizieren und verwenden:

```
//Initialize TIA Portal instance
var tiaPortalUi = new TiaPortal(TiaPortalMode.WithUserInterface);

//Open the project
var project = tiaPortalUi.Projects.Open(new FileInfo(projectPath));

//Enable secure mode for the PLC

//For R/H type handling
var onlineProvider = device.GetService<RHOnlineProvider>();
onlineProvider.Configuration.OnlineLegitimation += Configuration_OnlineLegitimation;

private void Configuration_OnlineLegitimation(OnlineConfiguration onlineConfiguration)
{
    var tlsCommunication = onlineConfiguration as TlsVerificationConfiguration;
    if(tlsCommunication == null) return;
    tlsCommunication.CurrentSelection = TlsVerificationConfigurationSelection.Trusted;
}

//For any other CPU type
var onlineProvider = project.Devices[0].DeviceItems[1].GetService<OnlineProvider>();
onlineProvider.Configuration.OnlineLegitimation += Configuration_OnlineLegitimation;

private void Configuration_OnlineLegitimation(OnlineConfiguration onlineConfiguration)
{
    var tlsCommunication = onlineConfiguration as TlsVerificationConfiguration;
    if(tlsCommunication == null) return;
    tlsCommunication.CurrentSelection = TlsVerificationConfigurationSelection.Trusted;
}
```

5.19.3.10 Import/Export von Testfällen mit Openness APIs

Verwendete Methoden

Für den Import und Export von Testfällen werden die folgenden Methoden verwendet:

Name der Methode	Beschreibung
SaveToFile()	Für den Export des Testfalls
LoadFromFile()	Für den Import des Testfalls

Speichern eines Testfalls in einer externen Textdatei

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.

Applikation

Mit der Methode `SaveToFile()` kann ein Testfall in einen ausgewählten Pfad exportiert werden.

Signatur

```
public void SaveToFile(FileInfo path);
```

Die folgende Tabelle zeigt die benötigten Parameter für die Methode:

Name des Parameters	Beschreibung
<code>FileInfo</code>	Vollständiger Pfad des Testfalls, der gespeichert werden soll

Programmcode

Hier sind `motorFBTestCase` und `motorFBTestCase1` Testfälle, die in "MyProject" zur Verfügung stehen.

```
motorFBTestCase.SaveToFile(exportFile);  
motorFBTestCase1.SaveToFile(new FileInfo(@"D:\Temp\TSTestcases2.tat"));
```

Testfälle aus externer Textdatei in Projekt laden

Voraussetzung

- TIA Portal Openness ist mit dem TIA Portal verbunden.

Applikation

Mit der Methode `LoadFromFile()` für `TestCaseComposition` kann ein Testfall aus der ausgewählten Datei importiert werden.

Signatur

```
public IEnumerable<TestCase> LoadFromFile (FileInfo path,  
ImportOptions loadOptions, TCLoadOptions tcLoadOptions);
```

Die folgende Tabelle zeigt die benötigten Parameter für die Methode:

Name des Parameters	Beschreibung
<code>FileInfo</code>	Vollständiger Pfad des Testfalls, der gespeichert werden soll
<code>ImportOptions.None</code>	Erzeugt eine Exception, wenn ein Testfall mit diesem Namen bereits im Projekt existiert / wenn die Datei einen Testfall mit dem gleichen Namen mehrmals enthält
<code>ImportOptions.Override</code>	Überschreibt den Testfall im Projekt mit dem gleichen Namen
<code>tcLoadOptions.IgnoreInvalidObject</code>	Importiert den Testfall, wenn der Anwendungsbereich / die Eigenschaft in der Datei nicht gültig ist.
<code>tcLoadOptions.None</code>	Erzeugt eine Exception, wenn der Testfall mit ungültigem Anwendungsbereich / ungültigen Eigenschaften importiert wird

Programmcode

Hier ist `motorFBTestCases` eine Instanz einer `TestCaseComposition`.

Beispiele:

```
// Load test cases from above obtained file path to project
// Throw Exception if the test case with the same name exists already
// Import the test case with a invalid scope/properties

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSTestcases1.tat");
motorFBTestcases.LoadFromFile(LoadFile, ImportOptions.None,
TCLoadOptions.IgnoreInvalidObject);

// Load test cases from above obtained file path to project
// Throw Exception if the test case with the same name exists already
// Throw Exception if the test case created with a invalid scope/invalid properties

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSTestcases1.tat");
motorFBTestcases.LoadFromFile(LoadFile, ImportOptions.None, TCLoadOptions.None);

// Load test cases from above obtained file path to project
// Import test case with the same name exists already
// Import the test case with a invalid scope/properties

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSTestcases1.tat");
motorFBTestcases.LoadFromFile(LoadFile, ImportOptions.Override,
TCLoadOptions.IgnoreInvalidObject);

// Load test cases from above obtained file path to project
// Overwrite test case with the same name exists already in project
// Throw Exception if the test case created with a invalid scope /invalid properties

FileInfo LoadFile = new FileInfo(@"D:\Temp\TSTestcases1.tat");
motorFBTestcases.LoadFromFile(LoadFile, ImportOptions.Override, TCLoadOptions.None);
```

5.19.3.11 Mit Bibliotheken arbeiten**Einen Testfall in die Bibliotheken kopieren**

Testfälle aus dem Projekt können zur Speicherung in die Bibliotheken kopiert werden, damit sie wiederverwendet und abgeändert werden können. Sie müssen die Testfälle für eine externe Änderung nicht explizit exportieren/importieren.

Wird die Projektbibliothek im Schreib-Lese-Modus geöffnet, können Sie am Zielort eine MasterCopy von einer IMasterCopySource erstellen.

Referenz

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public MasterCopy Create (IMasterCopySource sourceObject);
```

Verwendete Methoden

Name der Methode	Beschreibung
Create()	Zum Kopieren eines Testfalls von einem Projekt in eine Projekt- bzw. globale Bibliothek

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um einen Testfall von einem Projekt in die Bibliothek zu kopieren:

```
using Siemens.Engineering;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;
...

// Retrieving test case to copy to libraries:
TestCase myTestCase = testSuite.ApplicationTestGroup.TestCases.Find("CalcTestCase");

// For copying a test case to project library master copy system folder:
MasterCopy masterCopyInSystemFolder =
tiaProject.ProjectLibrary.MasterCopyFolder.MasterCopies.Create(myTestCase );

// For copying a test case to project library master copy user folder:
MasterCopy masterCopyinUserFolder =
tiaProject.ProjectLibrary.MasterCopyFolder.Folders[x].MasterCopies.Create(myTestCase );
tiaPortal.GlobalLibraries.Open(new FileInfo(@"D:\Documents\.."), OpenMode.ReadWrite);

// For copying a test case to global library master copy system folder:
MasterCopy masterCopyInSystemFolder =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.MasterCopies.Create(myTestCase );

// For copying a test case to global library master copy user folder:
MasterCopy masterCopyinUserFolder =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.Folders[x].MasterCopies.Create(myTestCase );
```

Masterkopien in Projekte kopieren

Die API-Schnittstelle von TIA Portal Openness unterstützt das Kopieren von Masterkopien in Projekte mit Hilfe der Aktion `CreateFrom`. Die Aktion erstellt ein neues Objekt auf Grundlage der Masterkopie und platziert es in der Struktur, in der die Aktion aufgerufen wurde. Die Aktion versucht, den oder die neuen Testfälle mit demselben Namen wie die Quell-Masterkopie zu erstellen. Wenn ein solcher Name in `TestCaseComposition` bereits besteht, gibt das System dem neuen Testfall einen neuen Namen. Dann wird der neue Testfall kopiert zurückgegeben.

Referenz

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public TestCase CreateFrom (MasterCopy masterCopy);
```

Verwendete Methoden

Name der Methode	Beschreibung
<code>CreateFrom()</code>	Zum Kopieren von Masterkopien aus Projekt- bzw. globalen Bibliotheken in das Projekt

Programmcode

Sie können das folgende Programmcodebeispiel modifizieren und verwenden, um eine Masterkopie von einem Projekt in die Bibliothek zu kopieren:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.ApplicationTest;

...

// For accessing master copy object in global library to copy to project:
var masterCopyObjectinGL = tiaPortal.GlobalLibraries[0].MasterCopyFolder.MasterCopies[0];
var masterCopyObjectinUserFolderinGL =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.Folders[0].MasterCopies[1];

// To copy test case from global library system folder to project:
TestCase testCase1
=testSuite.ApplicationTestGroup.TestCases.CreateFrom(masterCopyObjectinGL);

// To copy test case from global library user folder to project:
TestCase testCase2
=testSuite.ApplicationTestGroup.TestCases.CreateFrom(masterCopyObjectinUserFolderinGL);

// For accessing master copy object in project library to copy to project:
var masterCopyObjectinPL =tiaProject.ProjectLibrary.MasterCopyFolder.MasterCopies[0];
var masterCopyObjectinUserFolderinPL
=tiaProject.ProjectLibrary.MasterCopyFolder.Folders[0].MasterCopies[0];

// To copy test case from project library system folder to project:
TestCase testCase3
=testSuite.ApplicationTestGroup.TestCases.CreateFrom(masterCopyObjectinPL);

// To copy test case from project library user folder to project:
TestCase testCase4
=testSuite.ApplicationTestGroup.TestCases.CreateFrom(masterCopyObjectinUserFolderinPL);
```

Kopieren von Masterkopien (vom Typ TestCase) innerhalb von und zwischen Bibliotheken

Informationen zum Kopieren von Masterkopien (vom Typ TestCase) innerhalb von und zwischen Bibliotheken finden Sie in **TIA Portal Openness API** unter **Funktionen für Bibliotheken** im Punkt **Masterkopien kopieren** im Informationssystem.

5.19.4 Funktionen für Systemtest

5.19.4.1 Zugriff auf den Systemorder für den Systemtest

Programmcode

Mit den folgenden Eigenschaften können Sie auf den Systemordner eines Systemtests als Systemgruppe zugreifen:

```
using Siemens.Engineering;  
using Siemens.Engineering.TestSuite;  
using Siemens.Engineering.TestSuite.SystemTest;  
TestSuiteService testSuite = project.GetService<TestSuiteService>( );  
SystemTestSystemGroup stFolder = testSuite.SystemTestGroup;
```

'project' ist hier eine Instanz des TIA Portal-Projekts, die in die Openness-Anwendung geladen wurde.

Referenz

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse für den Systemtest:

Eigenschaftsname	Datentyp	Art des Zugriffs
SystemTestGroup	TestSuite.SystemTest.SystemTestSystemGroup	Lesen

5.19.4.2 Zugriff auf TestCaseComposition

Programmcode

Mit der folgenden Eigenschaft werden alle verfügbaren Systemtestfälle im Projekt aufgerufen:

```
SystemTestCaseComposition listTestCases = testSuite.SystemTestGroup.SystemTestCases;
```

Referenz

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse für den Systemtest:

Eigenschaftsname	Datentyp	Art des Zugriffs
SystemTestCases	TestSuite.SystemTestCaseComposition	Lesen/Schreiben

5.19.4.3 Zugriff auf Systemtestfall

Beschreibung

Testfälle in einem TIA Portal-Projekt können auf die folgenden zwei verschiedenen Arten in die Openness-Anwendung geladen werden:

1. Indizierter Zugriff auf SystemTestCaseComposition
2. Verwendung von Find() method in SystemTestCaseComposition

Programmcode

Mit den folgenden Eigenschaften können Sie auf einen einzelnen Testfall im Projekt zugreifen und einen bestimmten Testfall aus dem Projekt laden:

```
SystemTestCase motorFCTestcase = testSuite.SystemTestGroup.SystemTestCases[1];  
SystemTestCase testCase = testSuite.SystemTestGroup.SystemTestCases.Find("TCMot");
```

Referenz

Die folgende Tabelle beschreibt die Eigenschaften der Klasse für den Systemtest:

Eigenschaftsname	Datentyp	Art des Zugriffs
SystemTestCases	TestSuite.SystemTestCaseComposition	Lesen/Schreiben

Methoden:

```
public SystemTestCase Find(string Name)
```

5.19.4.4 Zugriff auf Namen des Systemtestfalls

Zugriff auf Namen des Systemtestfalls

Mit der folgenden Eigenschaft können Sie auf den Namen des Testfalls zugreifen:

```
string testCaseName = testSuite.SystemTestGroup.SystemTestCases[0].Name;
```

Umbenennen des ausgewählten Testfalls mit einem gültigen Namen

Mit der folgenden Eigenschaft können Sie den ausgewählten Testfall mit einem gültigen Namen umbenennen:

```
public string Name { get; set; }
```

Hinweis

Mit Hilfe der Eigenschaft "Name" können Sie die folgenden möglichen Aufgaben ausführen:

- Den Namen des existierenden Testfalls lesen
 - Den Namen des existierenden Testfalls ändern
-

Referenz

Die folgende Tabelle beschreibt die Eigenschaft der Klasse für den Systemtest:

Eigenschaftsname	Datentyp	Art des Zugriffs
Name	String	Lesen/Schreiben

Programmcode

Sie können den folgenden Programmcode ändern, um einen Testfall mit einem gültigen Namen umzubenennen.

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.SystemTest;
ProjectBase tiaProject = ... ;
TestSuiteService testSuite = tiaProject.GetService<TestSuiteService>();
SystemTestCaseExecutor systemTestService =
testSuite.SystemTestGroup.GetService<SystemTestCaseExecutor>();
//Retrieve single test case
SystemTestCase myTestCase1 = testSuite.SystemTestGroup.SystemTestCases[0];
//Read the name of a test case
String myTestCase_name = myTestCase1.Name;
```

5.19.4.5 Zugriff auf den Bereich eines Systemtestfalls

Zugriff auf den Bereich eines vorhandenen Testfalls

Der ausgewählte Bereich eines Testfalls kann mit den folgenden Testfalleigenschaften abgerufen werden

Die Syntax ist wie folgt:

```
public string OPCUAServerAddress { get; }
public DirectoryInfo OPCUAServerInterfaceFolderPath { get; }
public ServerInterfaces OPCUAServerInterfaceType { get; }
```

In der folgenden Tabelle werden die Parameter beschrieben:

Name	Datentyp	Beschreibung
OPCUAServerAddress	String	Für den Testfall konfigurierte OPC UA Serveradresse
OPCUAServerInterfaceFolderPath	DirectoryInfo	Angabe zum Verzeichnis für die zugeordneten Server-Schnittstellendateien
OPCUAServerInterfaceType	Siemens.Engineering.TestSuite.SystemTest.ServerInterfaces	Konfigurierter Serverschnittstellentyp (Benutzerdefiniert / Standard-SIMATIC / SiOME-Begleitspezifikation).

Programmcode

Sie können den folgenden Beispiel-Programmcode ändern und verwenden, um auf den Bereich des Systemtestfalls zuzugreifen:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.SystemTest;
ProjectBase tiaProject = ... ;
TestSuiteService testSuite = tiaProject.GetService<TestSuiteService>();
SystemTestCase tcForModule1 = testSuite.SystemTestGroup.SystemTestCases[0];
//Retrieve the OPC UA server address configured for a test
casestring tcForModule1OPCUAServeraddress = tcForModule1.OPCUAServerAddress;
//Retrieve the directory information of the OPC UA server interfaces mapped for a test
caseDirectoryInfo tcForModule1ServerInterfaces =
tcForModule1.OPCUAServerInterfaceFolderPath;
//Retrieve the full folder path of the OPC UA server interfaces mapped for a test
casestring tcForModule1ServerInterfacesPath = tcForModule1ServerInterfaces.FullName;
//Validate the folder path of the OPC UA server interfaces mapped for a test
caseDirectoryInfo directoryInfo=new DirectoryInfo(@"D:\TestSuite\SystemTest\PACKMLFiles");
Assert.AreEqual(systemTestCase.OPCUAServerInterfaceFolderPath, directoryInfo);
// Retrieve the OPC UA server interface selected for a test case
ServerInterface tcForModule1ServerInterfaceType = tcForModule1.OPCUAServerInterfaceType;
```

5.19.4.6 Bereich eines Systemtestfalls ändern

Bereich für Systemtestfall einstellen

Mit der folgenden API kann der Bereich eines vorhandenen Systemtestfalls geändert werden.

Die Methoden haben die folgende Syntax:

```
public void SetScope ( string OPCUAServerAddress, ServerInterfaces  
OPCUAServerInterfaceType, DirectoryInfo OPCUAServerInterfaceFolderPath );
```

Hier kann der Bereich mit Hilfe von OPCUAServerAddress, OPCUA ServerInterfaceType und OPCUA ServerInterfaceFolderPath festgelegt werden.

Da der Ordnerpfad für die Ausführung des Testfalls nicht zwingend benötigt wird, kann die folgende Syntax verwendet werden, um den Bereich ohne Parameter "OPCUAServerInterfaceFolderPath" zu ändern.

```
public void SetScope ( string OPCUAServerAddress, ServerInterfaces  
OPCUAServerInterfaceType);
```

In der folgenden Tabelle werden die Parameter beschrieben:

Name	Datentyp	Beschreibung
OPCUAServerAddress	String	Für den Testfall konfigurierte OPC UA Serveradresse
OPCUAServerInterfaceFolderPath	DirectoryInfo	Angabe zum Verzeichnis für die zugeordneten Server-Schnittstellendateien
OPCUAServerInterfaceType	Siemens.Engineering.TestSuite.System-Test.ServerInterfaces	Konfigurierter Serverschnittstellentyp (Benutzerdefiniert / Standard-SIMATIC / SiOME-Begleitspezifikation).

Hinweis

Sie können ferner den Bereich eines Testfalls auch einem anderen Testfall zuordnen. Weitere Informationen finden Sie im folgenden Programmcode.

Hinweis

Der Testfall wird automatisch aktualisiert, wenn der Bereich des Testfalls geändert wird. Weitere Informationen finden Sie unter Systemtestfallbereich aktualisieren

Programmcode

Sie können den folgenden Beispiel-Programmcode ändern und verwenden:

```
using Siemens.Engineering;
.....
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.SystemTest;

// Accessing System test case
SystemTestCase tcModule1 =
tiaProject.GetService<TestSuiteService>().SystemTestGroup.SystemTestCases[1];

// OPC UA server address
string targetPLCasOPCUAServer = "opc.tcp://opcua.demo-setscope.com:51210/UA/SampleServer";

//Folder path to OPC UA server interfaces stored
DirectoryInfo tcModule1ServerInterafcesPath = new
DirectoryInfo(@"D:\Machines\ModuleTest\ServreInterafces");

//Setting the scope of a selected test case
tcModule1.SetScope(targetPLCasOPCUAServer ,ServerInterfaces.StandardSIMATIC ,
tcModule1ServerInterafcesPath);

*****
Assigning the scope of one test case to another test case
*****

SystemTestCase tcForModule3 = testSuite.SystemTestGroup.SystemTestCases[3];
SystemTestCase tcForModule4 = testSuite.SystemTestGroup.SystemTestCases[4];

//Retrieve the OPC UA server address configured for a "tcForModule3"

string tcForModule3OPCUAServeraddress = tcForModule3.OPCUAServerAddress;

//Retrieve the directory information of the OPC UA server interfaces mapped for
a "tcForModule3"

DirectoryInfo tcForModule3ServerInterfaces = tcForModule3.OPCUAServerInterfaceFolderPath;

// Retrieve the OPC UA server interface selected for a "tcForModule3"

ServerInterfcaes tcForModule3ServerInterfaceType = tcForModule3.OPCUAServerInterfaceType;

//Setting the scope of a "tcForModule4"
tcForModule4.SetScope(tcForModule3OPCUAServeraddress, tcForModule3ServerInterfaceType,
tcForModule3ServerInterfaces);
```

Fehlermeldungen während der Änderung des Bereichs

Die folgenden Anwendungsszenarien mit den entsprechenden Fehlermeldungen sind nachstehend erläutert.

Anwendungsszenarien	Fehlermeldungen
"OPCUAServerAddress" ist nicht im richtigen Format angegeben, d. h. opc.tcp://server:port/path	Die angegebene "OPCUAServerAddress" ist ungültig. Verwenden Sie das Format 'opc.tcp://server:port/path'
Für "OPCUAServerAddress" wird ein Null-Wert eingegeben	"OPCUAServerAddress" darf nicht null oder leer sein. Verwenden Sie das Format 'opc.tcp://server:port/path'
Es wird versucht, den Bereich eines Testfalls in einem anderen Projekt zu aktualisieren.	Fehlermeldung aus dem Openness-Framework
Keine passenden Dateien im angegebenen "OPCUAServerInterfaceFolderPath" vorhanden	Der angegebene "OPCUAServerInterfaceFolderPath" enthält keine gültigen Server-Schnittstellendateien
Der in "OPCUAServerInterfaceFolderPath" angegebene Ordnerpfad ist nicht verfügbar	Verzeichnis '<Laufwerk_Buchstabe>:\' nicht gefunden
Der in "OPCUAServerInterfaceFolderPath" angegebene Ordnerpfad enthält unzulässige Zeichen	Unzulässige Zeichen in der Pfadangabe
Der in "OPCUAServerInterfaceFolderPath" angegebene Ordnerpfad ist nicht zulässig	Datei '<FolderPath>' kann nicht gelesen werden
Der in "OPCUAServerInterfaceFolderPath" angegebene Ordnerpfad ist ein spezifischer/relativer Pfad	Das Argument '<path>' darf kein spezifischer Pfad sein
Der in "OPCUAServerInterfaceFolderPath" angegebene Ordnerpfad enthält zusätzliche Leerzeichen (d. h. führende und nachgestellte Leerzeichen)	Das Argument '<path>' enthält zusätzliche Leerzeichen; dies ist nicht erlaubt.
Der in "OPCUAServerInterfaceFolderPath" angegebene Ordnerpfad ist ein trennbarer/entfernbarer Pfad	Verzeichnis G:\ für Pfad 'G:\\RemovableUSB_Disk' nicht gefunden
Der in "OPCUAServerInterfaceFolderPath" angegebene Ordnerpfad enthält mehr als 248 Zeichen	Der angegebene Pfad, Dateiname oder beide sind zu lang.

5.19.4.7 Ausführung von Systemtestfällen

Ausführung von Systemtestfällen

Die Ausführung von Testfällen kann bei der Openness-Anwendung auf drei Arten geschehen, wie nachstehend beschrieben:

1. An einem bestimmten Testfall
2. An einer Liste von Testfällen
3. An allen verfügbaren Testfällen im Systemtestordner

Referenz

Signatur

```
public SystemTestCaseExecutor GetService<SystemTestCaseExecutor>();
public TestResults Run(SystemTestCase myTestCase);
public TestResults Run(new IEnumerable<SystemTestCase>() {myTestCase1,myTestCase2,.. });
public TestResults Run(SystemTestGroup allTestCases);
```

Die folgenden Attribute werden von TestResults unterstützt:

Attributbezeichnung	Datentyp	Art des Zugriffs	Beschreibung
Messages	Siemens.Engineering.Test-Suite.TestResults-MessgeComposition	Lesen	Zusammensetzung der Rückmeldung
Parent	IEngineeringObject	Lesen	Liefert das übergeordnete Element des Containers zurück
State	Siemens.Engineering.Test-Suite.TestResultsState	Lesen	Endzustand in einer Testergebnisrückmeldung mit den möglichen Werten: Warning, Error und Success
Warning-Count	Int	Lesen	Anzahl der Warnungen in einer Testergebnisrückmeldung
ErrorCount	Int	Lesen	Anzahl der Fehler in einer Testergebnisrückmeldung
Description	String	Lesen	Beschreibung oder Inhalt einer Testergebnisrückmeldung
DateTime	DateTime	Lesen	Datum und Uhrzeit einer Testergebnisrückmeldung
Path	String	Lesen	Pfad zu einer Testergebnisrückmeldung

Verwendete Methoden:

Name der Methode	Beschreibung
Run()	Zur Ausführung der ausgewählten Testfälle

Hinweis

Format des Zeitstempels

Alle Zeitstempel sind in UTC angegeben. Wenn Sie sich die Ortszeit anzeigen lassen möchten, verwenden Sie `DateTime.ToLocalTime()`.

Programmcode

Sie können das folgende Programmcodebeispiel abwandeln und verwenden:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.SystemTest;
ProjectBase tiaProject = ... ;
TestSuiteService testSuite = tiaProject.GetService<TestSuiteService>();
SystemTestCaseExecutor systemTestService =
testSuite.SystemTestGroup.GetService<SystemTestCaseExecutor>();
//To execute specific test case:
//Retrieve single test case:
//Test case at 0th index is available in the project:
SystemTestCase myTestCase1 = testSuite.SystemTestGroup.SystemTestCases[0];
//TestCase_10 is available in the project:
SystemTestCase myTestCase2 = testSuite.SystemTestGroup.SystemTestCases.Find("TestCase_10");
//Execute above retrieved test cases:TestResults TestCaseResults1 =
systemTestService.Run(myTestCase1);
TestResults TestCaseResults2 = systemTestService.Run(myTestCase2);
//To execute list of test cases://Retrieve list of test cases to execute:
IEnumerable<SystemTestCase> TestCaseList = new List<SystemTestCase>()
{ myTestCase1,myTestCase2 };
//Execute above retrieved list of test cases:
TestResults TestCasesResults3 = systemTestService.Run(TestCaseList);
//To execute all test cases in SystemTestGroup:
//Retrieve list of test cases to execute:
SystemTestSystemGroup systemTestFolder = testSuite.SystemTestGroup;
//Execute all available test cases in SystemTestFolder:
TestResults TestCasesResults4 = systemTestService.Run(systemTestFolder);
```

Der Anwender kann auf die obigen Testergebnisse zugreifen und mithilfe des folgenden Codes durch die TestResults-Meldungen iterieren:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.SystemTest;
...
//Retrieve test results of single test case execution
WriteTestResults(TestCaseResults1);
//Retrieve test results of list of test cases execution
WriteTestResults(TestCasesResults3);
//Retrieve test results of test cases execution available in "SystemTestGroup"
WriteTestResults(TestCasesResults4);
...
private void WriteTestResults (TestResults result)
{
    Console.WriteLine("State:" + result.State);
    Console.WriteLine("Warning Count:" + result.WarningCount);
    Console.WriteLine("Error Count:" + result.ErrorCount);
    RecursivelyWriteMessages(result.Messages);
}
private void RecursivelyWriteMessages(TestResultsMessageComposition messages, string
indent= "")
{
    indent += "\t";
    foreach (TestResultsMessage message in messages)
    {
        Console.WriteLine(indent + "Path: " + message.Path);
        Console.WriteLine(indent + "DateTime: " + message.DateTime);
        Console.WriteLine(indent + "State: " + message.State);
        Console.WriteLine(indent + "Description: " + message.Description);
        Console.WriteLine(indent + "Warning Count: " + message.WarningCount);
        Console.WriteLine(indent + "Error Count: " + message.ErrorCount);
        RecursivelyWriteMessages(message.Messages, indent);
    }
}
```

5.19.4.8 Import/Export von Systemtestfällen mit Openness APIs

Verwendete Methoden

Für den Import und Export der Testfälle werden die folgenden Methoden verwendet:

Name der Methode	Beschreibung
SaveToFile()	Für den Export des Systemtestfalls
LoadFromFile()	Für den Import der Systemtestfälle

Speichern eines Testfalls in einer externen Textdatei

Voraussetzung

TIA Portal Openness ist mit dem TIA Portal verbunden.

Anwendung

Mit der Methode `SaveToFile()` kann ein Testfall in einen ausgewählten Pfad exportiert werden.

Signatur

```
public void SaveToFile(FileInfo path);
```

Die folgende Tabelle zeigt die benötigten Parameter für die Methode:

Name des Parameters	Beschreibung
FileInfo	Vollständiger Pfad des zu speichernden Systemtestfalls

Programmcode

Hier sind `motorFBTestCase` und `motorFBTestCase1` Testfälle, die in "MyProject" zur Verfügung stehen.

```
motorFBTestCase.SaveToFile(exportFile);
motorFBTestCase1.SaveToFile(new FileInfo(@"D:\Temp\TSTestcases2.tst"));
```

Testfälle aus externer Textdatei in Projekt laden**Voraussetzung**

TIA Portal Openness ist mit dem TIA Portal verbunden.

Anwendung

Mit der Methode `LoadFromFile()` für `SystemTestCaseComposition` kann ein Testfall aus der ausgewählten Datei importiert werden.

Signatur

```
public IList<SystemTestCase> LoadFromFile (FileInfo
path, ImportOptions loadOptions, TcLoadOptions tcLoadOptions);
```

Die folgende Tabelle zeigt die benötigten Parameter für die Methode:

Name des Parameters	Beschreibung
FileInfo	Vollständiger Pfad des zu speichernden Testfalls
ImportOptions.None	Erzeugt eine Exception, wenn ein Testfall mit dem gleichen Namen bereits im Projekt existiert
ImportOptions.Override	Überschreibt den Testfall im Projekt mit dem gleichen Namen
tcLoadOptions.IgnoreInvalidObject	Importiert den Testfall durch den Testfallbereich (ausgewählte OPC UA-Serveradresse und Schnittstelle / Eigenschaften in der Datei ungültig/fehlen)
tcLoadOptions.None	Erzeugt eine Exception, wenn der Testfall mit ungültiger/ fehlender OPC UA-Serveradresse importiert wird, und auch dann, wenn die Datei <code>Interface_Selected</code> enthält / Testfalleigenschaften ungültig sind

Programmcode

Hier ist motorFBTestCases eine Instanz einer SystemTestCaseComposition.

Beispiele:

```
// Load test cases from above obtained file path to project
// Throw Exception if the test case with the same name exists already
// Import the test case with invalid scope/properties
FileInfo LoadFile = new FileInfo(@"D:\Temp\TSTestcases1.tst");
motorFBTestcases.LoadFromFile(LoadFile,
ImportOptions.None, TCLoadOptions.IgnoreInvalidObject);
// Load test cases from above obtained file path to project
// Throw Exception if the test case with the same name exists already
// Throw Exception if the test case created with invalid/not available scope/invalid
properties
FileInfo LoadFile = new FileInfo(@"D:\Temp\TSTestcases1.tst");
motorFBTestcases.LoadFromFile(LoadFile, ImportOptions.None, TCLoadOptions.None);
// Load test cases from above obtained file path to project
// Import test case with the same name exists already
// Import the test case with invalid scope/properties
FileInfo LoadFile = new FileInfo(@"D:\Temp\TSTestcases1.tst");
motorFBTestcases.LoadFromFile(LoadFile,
ImportOptions.Override, TCLoadOptions.IgnoreInvalidObject);
// Load test cases from above obtained file path to project
// Overwrite test case with the same name exists already in project
// Throw Exception if the test case created with invalid/not available scope/invalid
properties
FileInfo LoadFile = new FileInfo(@"D:\Temp\TSTestcases1.tst");
motorFBTestcases.LoadFromFile(LoadFile, ImportOptions.Override, TCLoadOptions.None);
```

5.19.4.9 Arbeiten mit Bibliotheken**Einen Systemtestfall in die Bibliotheken kopieren**

Systemtestfälle aus einem Projekt lassen sich zur Speicherung in die Bibliotheken kopieren, damit sie wiederverwendet und abgeändert werden können. Sie müssen dann die Systemtestfälle für externe Änderungen nicht explizit exportieren/importieren.

Wird die Projektbibliothek im Schreib-Lese-Modus geöffnet, können Sie am Zielort eine Masterkopie (MasterCopy) von einer IMasterCopySource erstellen.

Referenz

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public MasterCopy Create (IMasterCopySource sourceObject);
```

Verwendete Methoden

Name der Methode	Beschreibung
Create()	Zum Kopieren eines Systemtestfalls von einem Projekt in eine Projekt- bzw. globale Bibliothek

Programmcode

Sie können das folgende Programmcodebeispiel ändern und verwenden, um einen Testfall von einem Projekt in die Bibliothek zu kopieren:

```
using Siemens.Engineering;
...
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.SystemTest;
...

// Retrieving a system test case to copy to libraries:
SystemTestCase mySystemTestCase =
testSuite.SystemTestGroup.SystemTestCases.Find("CalcTestCase");

// For copying a system test case to project library master copy system
folder:
MasterCopy masterCopyInSystemFolder =
tiaProject.ProjectLibrary.MasterCopyFolder.MasterCopies.Create(mySystemTes
tCase );

// For copying a system test case to project library master copy user folder:
MasterCopy masterCopyinUserFolder =
tiaProject.ProjectLibrary.MasterCopyFolder.Folders[0].MasterCopies.Create(
mySystemTestCase );
tiaPortal.GlobalLibraries.Open(new FileInfo(@"D:\Documents\.."),
OpenMode.ReadWrite);

// For copying a test case to global library master copy system folder:
MasterCopy masterCopyInSystemFolder =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.MasterCopies.Create(mySystem
TestCase );

// For copying a test case to global library master copy user folder:
MasterCopy masterCopyinUserFolder =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.Folders[0].MasterCopies.Crea
te(mySystemTestCase );
```

Masterkopien in Projekte kopieren

Die API-Schnittstelle von TIA Portal Openness unterstützt das Kopieren von Masterkopien in Projekte mithilfe der Aktion `CreateFrom`. Die Aktion erstellt ein neues Objekt auf Grundlage der Masterkopie und platziert es in der Struktur, in der die Aktion aufgerufen wurde. Die Aktion versucht, den oder die neuen Systemtestfälle mit demselben Namen wie die Quell-Masterkopie zu erstellen. Wenn ein solcher Name in `SystemTestCaseComposition` bereits besteht, gibt das System dem neuen Testfall einen neuen Namen. Dann wird der neue Systemtestfall kopiert zurückgegeben.

Referenz

Die folgende Tabelle beschreibt die Syntax der Klasse:

```
public SystemTestCase CreateFrom (MasterCopy masterCopy);
```

Verwendete Methoden

Name der Methode	Beschreibung
<code>CreateFrom()</code>	Zum Kopieren von Masterkopien aus Projekt- bzw. globalen Bibliotheken in das Projekt

Programmcode

Sie können das folgende Programmcodebeispiel ändern und verwenden, um eine Masterkopie von einem Projekt in die Bibliothek zu kopieren:

```
using Siemens.Engineering;
using Siemens.Engineering.TestSuite;
using Siemens.Engineering.TestSuite.SystemTest;

...

// For accessing master copy object in global library to copy to project:
var masterCopyObjectinGL =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.MasterCopies[0];
var masterCopyObjectinUserFolderinGL =
tiaPortal.GlobalLibraries[0].MasterCopyFolder.Folders[0].MasterCopies[1];

// To copy system test case from global library system folder to project:
SystemTestCase SystemtestCase1
=testSuite.SystemTestGroup.SystemTestCases.CreateFrom(masterCopyObjectinGL
);

// To copy test case from global library user folder to project:
SystemTestCase SystemtestCase2
=testSuite.SystemTestGroup.SystemTestCases.CreateFrom(masterCopyObjectinUs
erFolderinGL);

// For accessing master copy object in project library to copy to project:
var masterCopyObjectinPL
=tiaProject.ProjectLibrary.MasterCopyFolder.MasterCopies[0];
var masterCopyObjectinUserFolderinPL
=tiaProject.ProjectLibrary.MasterCopyFolder.Folders[0].MasterCopies[0];

// To copy system test case from project library system folder to project:
SystemTestCase SystemtestCase3
=testSuite.SystemTestGroup.SystemTestCases.CreateFrom(masterCopyObjectinPL
);

// To copy systemtest case from project library user folder to project:
SystemTestCase SystemtestCase4
=testSuite.SystemTestGroup.SystemTestCases.CreateFrom(masterCopyObjectinUs
erFolderinPL);
```

Kopieren von Masterkopien (vom Typ SystemTestCase) innerhalb von und zwischen Bibliotheken

Informationen zum Kopieren von Masterkopien (vom Typ SystemTestCase) innerhalb von und zwischen Bibliotheken finden Sie in **TIA Portal Openness API** unter **Funktionen für Bibliotheken** im Punkt **Masterkopien kopieren** im Informationssystem.

5.19.5 Exceptions

Behandlung von Exceptions

Exceptions beim Zugriff auf die Test Suite über TIA Portal Openness API

Während der Ausführung einer Test Suite-Anwendung über die TIA Portal Openness API werden alle auftretenden Fehler als Exceptions gemeldet. Diese Exceptions enthalten Informationen, die bei der Beseitigung der aufgetretenen Fehler helfen.

Es wird zwischen zwei Arten von Exceptions unterschieden:

- Recoverable (Siemens.Engineering.EngineeringException)
Mit dieser Exception haben Sie ohne Unterbrechung weiterhin Zugang zum TIA Portal. Alternativ können Sie die Verbindung zum TIA Portal abbrechen.
EngineeringExceptions umfassen die folgenden Typen:
 - Exceptions bei der Ausführung von Testfällen (SupportSimulationNotEnabledException), falls die Eigenschaft Simulation unterstützen im Projekt nicht aktiviert ist
 - Exception wegen ungültiger OPCUA-Server-Adresse (OPCUAServerAddressNotValidException) - Diese Exception tritt auf, wenn die ausgeführten Testfälle eine ungültige OPC UA-Serveradresse haben.
 - Exception wegen nicht gefundener Lizenz (LicenseNotFoundException) - Diese Exception tritt auf, wenn keine gültige Lizenz für die Test Suite verfügbar ist
- NonRecoverable (Siemens.Engineering.NonRecoverableException)
Diese Exception schließt das TIA Portal und die Verbindung zum TIA Portal wird getrennt. Sie müssen das TIA Portal mit Hilfe der Anwendung TIA Portal Openness neu starten.

5.20 Funktionen für SINUMERIK 840D sl

5.20.1 Einführung

Über TIA Portal-Openness automatisieren Sie das Engineering und steuern das TIA Portal über ein von Ihnen selbst erstelltes Programm.

Für dieses selbst erstellte Programm finden Sie in dieser Hilfe viele Informationen und Beispielcodes. Auch für die TIA Portal-Anwendung "SINUMERIK" können Sie eigene Programme zusammenstellen und anwenden.

Weitere Informationen

Bevor Sie für SINUMERIK aus den nachfolgend gelisteten Beispielcodes ein eigenes Programm zusammenstellen, beachten Sie bitte die übergreifenden Informationen zu Openness, die Sie unter folgenden Stichworten in dieser Hilfe finden:

- Voraussetzungen für TIA Portal Openness
- TIA Portal Openness installieren

- Auf das TIA Portal zugreifen
- TIA Portal Openness-Objektmodell
- Programmierschritte

5.20.2 Typkennung- Kennzeichnung der Komponenten

Jede SINUMERIK-Komponente besitzt eine eindeutige Nummer, die als Typkennung (Typidentifizier) bezeichnet wird. Im Openness-Programmcodex können Sie die Typkennung nutzen, um eine Komponente eindeutig zu identifizieren und zu benennen. Beispielsweise lautet die Typkennung für SINUMERIK 840D sl NCU 710.3 PN "Artikel-Nr. :6FC5 371-0AA30-0Axx/Vy.z".

Die Typkennung sehen Sie beim Anlegen Ihres Geräts im Dialog "Neues Gerät hinzufügen" sowie in der tabellarischen Geräteübersicht.

Sie können die Typkennung in Ihre Openness-Anwendung kopieren.

Die Typkennung wird in Openness als Aktualparameter beim Aufruf einer Methode, z. B. der `CreateWithItem()`-Methode, erwartet.

Anzeige des Typidentifiers in SINUMERIK aktivieren

1. Wählen Sie in der Projektansicht das Menü "Extras > Einstellungen".
Der Konfigurationsbereich "Einstellungen" wird geöffnet.
2. Wählen Sie in der Sekundärnavigation den Eintrag "Hardware-Konfiguration".
3. Aktivieren Sie die Option "Anzeige des Type Identifiers für Geräte und Module aktivieren".
Die Anzeige des Typidentifiers ist nun aktiv.

5.20.3 Grundlagen

Übersicht

Folgende SINUMERIK-Geräte können Sie in TIA Portal Openness anlegen:

- NCU
- NX-Baugruppe
- ADI4

Zum Anlegen der SINUMERIK-Geräte verwenden Sie die Methode `CreateWithItem()` der `Devices Collection`.

Hinweis

Alle integrierten Subkomponenten einer SINUMERIK-NCU wie PLC, NCK, CP, HMI und SINAMICS Integrated, werden unterlagert auf der gleichen Ebene automatisch angelegt.

Besonderheiten beim Anlegen der SINUMERIK-Geräte

Die Namen der Baugruppenträger entsprechen den NCU-Typen und sind in der Oberfläche schreibgeschützt. Die Bezeichnungen der Baugruppenträger müssen Sie auch in TIA Portal Openness verwenden.

Verwenden Sie die folgenden Standardnamen der Baugruppenträger für die Erstellung eines Geräts über eine Geräteelement-Typkennung mit der `Device CreateWithItem`-Methode:

SINUMERIK NCU	Baugruppenträger-Standardname
SINUMERIK 840D sl NCU 710.3 PN	NCU 710.3 PN
SINUMERIK 840D sl NCU 720.3 PN	NCU 720.3 PN
SINUMERIK 840D sl NCU 730.3 PN	NCU 730.3 PN
SINUMERIK 840D sl NCU 730.3 PN /319	NCU 730.3 PN /319

Hinweis

Alternativ können Sie die Parameternamen weglassen. Wenn der Name "Null" oder "String.Empty" ist, wird der Standardname verwendet.

Folgende Parameter können mit der Methode `CreateWithItem()` verwendet werden:

- `default name`,
z. B.
`project.Devices.CreateWithItem("OrderNumber:6FC5 371-0AA30-0AA0/V4.8", "NCU 710.3 PN", "TestDevice");`
- `null`,
z. B.
`project.Devices.CreateWithItem("OrderNumber:6FC5 371-0AA30-0AA0/V4.8", null, "TestDevice");`
- `string.Empty`,
z. B.
`project.Devices.CreateWithItem("OrderNumber:6FC5 371-0AA30-0AA0/V4.8", string.Empty, "TestDevice");`

Weitere Informationen zu Aufrufparametern der Methode `CreateWithItem()` finden Sie im Kapitel "Erstellen eines Geräts".

Die folgende Tabelle zeigt die Zuordnung der Geräte und ihrer Typkennungen:

SINUMERIK NCU	Typkennung
SINUMERIK 840D sl NCU 710.3 PN	Artikel-Nr. :6FC5 371-0AA30-0Axx/Vy.z
SINUMERIK 840D sl NCU 720.3 PN	Artikel-Nr. :6FC5 372-0AA30-0Axx/Vy.z
SINUMERIK 840D sl NCU 730.3 PN	Artikel-Nr. :6FC5 373-0AA30-0Axx/Vy.z
SINUMERIK 840D sl NCU 730.3 PN /319	Artikel-Nr. :6FC5 373-0AA31-0Axx/Vy.z

Beim Anlegen der SINUMERIK-Geräte sind Platzhalter in der Typkennung zulässig. Die Platzhalter tauschen Sie später gegen passende gerätespezifische Zeichen aus. Wenn Sie z. B. die Typkennung "OrderNumber:6FC5 372-0***0-0AA0/V4.8" eingeben, wird SINUMERIK 840D sl NCU 720.3 PN mit der Firmware-Version FW4.8 angelegt.

Klassifikation des Geräteelements

Jedes Gerät oder Geräteelement besitzt obligatorische Attribute, die gelesen und geschrieben werden. Weitere Informationen zu den Attributen, die in Openness unterstützt werden, finden Sie im Kapitel "Funktionen auf Geräteelementen".

Die Klassifikationsattribute des Geräteelements sind schreibgeschützt und in der Benutzeroberfläche des TIA Portals nicht sichtbar.

Die Klassifikationsattribute haben den Wert "DeviceItemClassification":

Wert des Klassifikationsattributs	Beschreibung
DeviceItemClassifications.None (0)	Keine Klassifikation.
DeviceItemClassifications.CPU (1)	Das Geräteelement ist eine CPU.
DeviceItemClassifications.HM (2)	Das Geräteelement ist ein Kopfmodul.

Wenn Sie den Wert des Geräteelements über die integrierte PLC abfragen, ist es für ein SINUMERIK-Gerät der Wert "CPU (1)".

Als Kopfmodule fungieren im Openness-Objektmodell die folgenden Komponenten:

- SINAMICS Integrated
- NX-Baugruppe
- ADI4

In allen anderen Fällen ist der Wert des Klassifikationsattributs "DeviceItemClassification" "None" (0).

Geräteelemente über die Eigenschaft "Kopfmodul" finden

Das folgende Beispiel zeigt, wie Sie die Geräteelemente mit der Eigenschaft "Kopfmodul" finden, bevor Sie z. B. ein Telegramm konfigurieren.

Sie können diese Eigenschaft nur lesen, aber nicht an Geräteelemente setzen.

Geräteelemente über die Eigenschaft "Kopfmodul" finden

```
foreach (Device device in project.Devices)
{
    foreach (DeviceItem deviceItem in device.DeviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.HM)
        {
            var driveObjectContainer = deviceItem.GetService<DriveObjectContainer>();
            // do something
        }
    }
}
```

Siehe auch

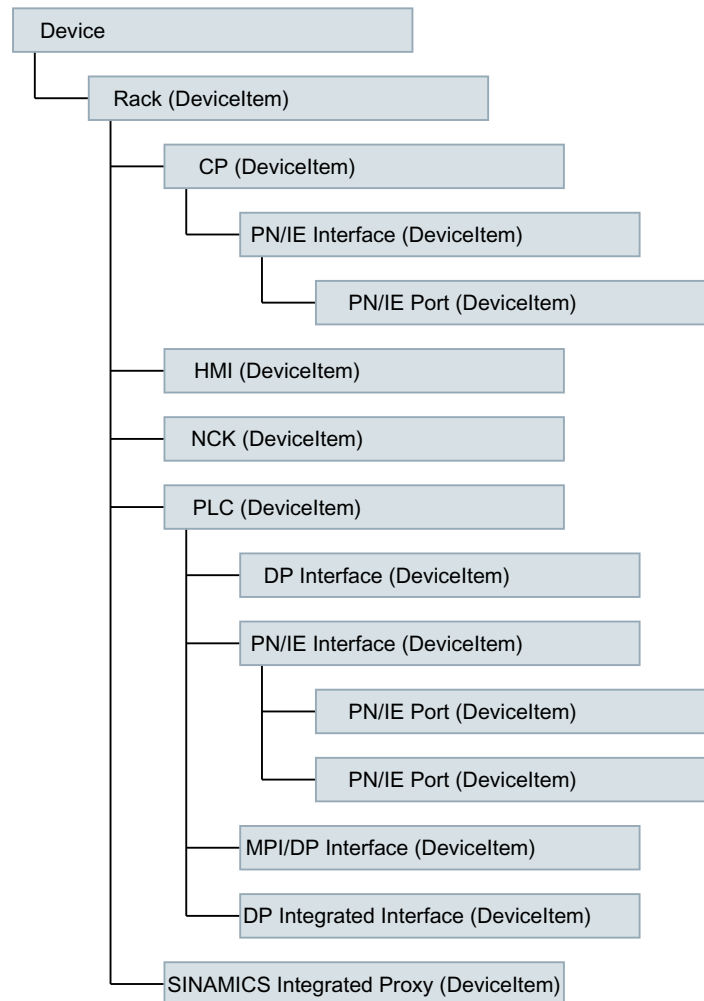
NX-Baugruppe anlegen (Seite 1084)

NCU anlegen (Seite 1083)

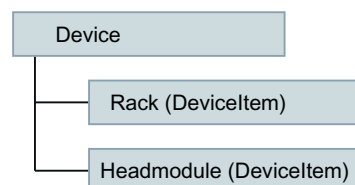
5.20.4 Objektmodell

Übersicht

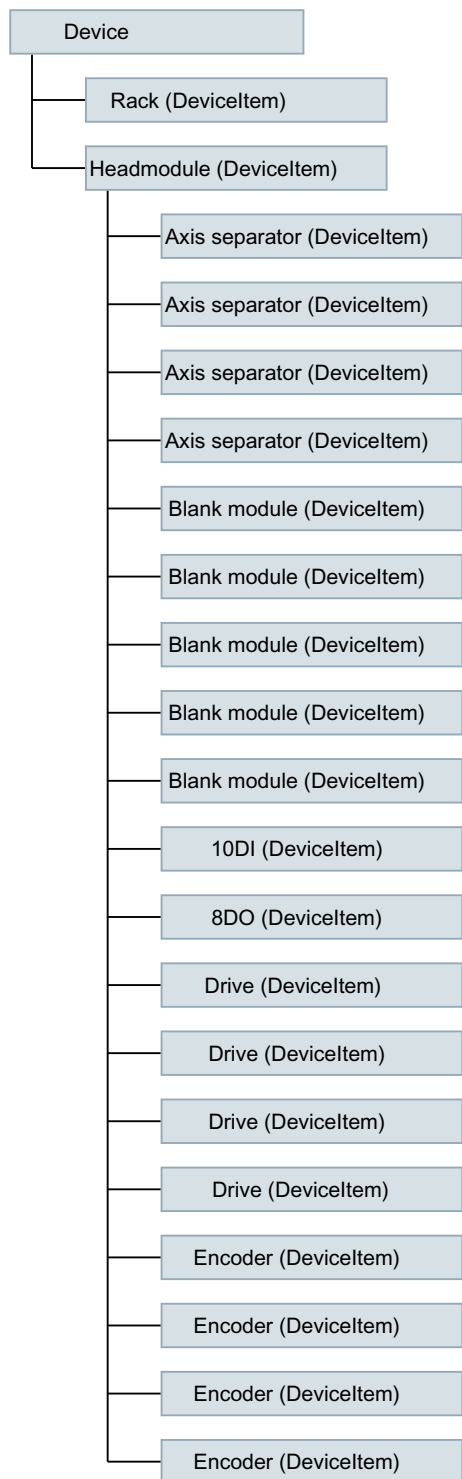
Das folgende Diagramm beschreibt die Objekte, die sich bei SINUMERIK 840D sl unter "Device" befinden:



Das folgende Diagramm beschreibt die Objekte, die sich bei SINUMERIK NX 10.3/15.3 unter "Device" befinden:



Das folgende Diagramm beschreibt die Objekte, die sich bei ADI4 unter "Device" befinden:



Weitere Informationen zu TIA Portal Openness-Objektmodell, finden Sie unter "TIA Portal Openness API".

5.20.5 Codebeispiel

5.20.5.1 Allgemein

Die folgenden Codebeispiele beschreiben die grundsätzliche Vorgehensweise für verschiedene Anwendungsfälle. Der Code ist nicht zwingend vollständig und kompilierbar.

5.20.5.2 Erste Schritte in SINUMERIK durchführen

- Verbinden Sie die TIA Portal Openness-Anwendung mit dem TIA Portal.
- Öffnen Sie Ihr Projekt.

Das folgende Beispiel zeigt, wie Sie ermitteln können, welche SINUMERIK Toolbox-Version installiert ist.

SINUMERIK Toolbox-Version ermitteln

```
using Siemens.Engineering;
if (tiaProcess.InstalledSoftware.Any(sw => sw.Name.Equals("SINUMERIK Toolbox") &&
sw.Version.Equals("V17")))
{
    Console.WriteLine("SINUMERIK Toolbox is available");
}
// "V17" is the current SINUMERIK version started at October 2021.
```

5.20.5.3 NCU anlegen

Mit der Methode `CreateWithItem()` der `Devices Collection` legen Sie eine SINUMERIK NCU an. Die SINUMERIK NCUs werden über Parameter der Methode spezifiziert. Das Format der Parameter ist nachfolgend beschrieben.

NCU anlegen

Format der Parameter für SINUMERIK NCU:

```
CreateWithItem(@"OrderNumber:mlfb/
FirmwareVersion/", "NameOfTheDevice", positionNumber)
```

Der Parameter `positionNumber` ist optional.

Das folgende Beispiel zeigt, wie Sie eine Steuerung SINUMERIK 840D sl NCU 720.3 PN anlegen.

SINUMERIK 840D sl NCU 720.3 PN anlegen

```
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
Project tiaproject= portal.Projects.Open("..."); //The path of the project

Device NCUDevice = tiaproject.Devices.CreateWithItem(@"OrderNumber:6FC5
372-0AA30-0AA0/4.8/", "NCU 720.3 PN", string.Empty, "TestDevice");
```

5.20.5.4 NX-Baugruppe anlegen

Mit der `Device.CreateWithItem`-Methode legen Sie eine NX-Baugruppe an. Anschließend verbinden Sie die NX-Baugruppe mit einer NCU.

Folgende Typkennungen werden für SINUMERIK NX-Baugruppen verwendet:

SINUMERIK NX-Baugruppe	Typkennung
SINUMERIK NX10.3	OrderNumber:6SL3 040-1NC00-0Axx/Vy.z
SINUMERIK NX15.3	OrderNumber:6SL3 040-1NB00-0Axx/Vy.z

Das folgende Beispiel zeigt, wie Sie eine SINUMERIK NX-Baugruppe anlegen.

NX-Baugruppe anlegen

```
project.Devices.CreateWithItem("OrderNumber:6SL3040-1NC00-0AA0/
V5.1", "MyNXDevice", "TestDevice");
```

Versionskompatibilität

Die Firmware-Version der NX-Baugruppe muss mit der Firmware-Version von SINAMICS Integrated übereinstimmen und mit der Firmware-Version der NCU kompatibel sein.

Die von SINAMICS Integrated abweichende NX Firmware-Version kann über Openness nicht zugeordnet werden.

Die folgende Tabelle zeigt die Versionskompatibilität für 840D sl:

NCU-Firmware (840D sl)	SINAMICS Integrated/NX-Firmware
V4.95	V5.2

5.20.5.5 NX-Baugruppe mit NCU verbinden

NX-Baugruppe mit einer NCU verbinden

Um eine NX-Baugruppe mit einer NCU über den Subnetztyp "ProfibusIntegrated" zu verbinden, muss der Dienst "NetworkInterface" über das Kopfmodul von NX geladen werden.

Das folgende Beispiel zeigt, wie Sie den Dienst "NetworkInterface" laden.

NetworkInterface-Dienst laden

```
foreach (Device device in project.Devices)
{
    foreach (DeviceItem deviceItem in device.DeviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.HM)
        {
            var networkInterface = deviceItem.GetService<NetworkInterface>();
            // do something
        }
    }
}
```


Das folgende Beispiel zeigt, wie Sie eine NX-Baugruppe über "ProfibusIntegrated" mit einer NCU verbinden:

NX-Baugruppe über ProfibusIntegrated verbinden

```
Subnet pbiSubnet = ...;  
Node node = networkInterface.Nodes.FirstOrDefault();  
node.ConnectToSubnet(pbiSubnet);
```

Die DP-Adresse wird in Openness über DRIVE-CLiQ Port der fest NX zugeordnet. Jedes Port-Label verfügt über eine feste DP-Integrated-Adresse.

Die DP-Adresse muss vor dem Verbinden mit der NCU der NX zugeordnet werden.

5.20.5.6 Safety Integrated aktivieren

Safety Integrated aktivieren

Mit TIA Portal Openness können Sie Safety Integrated (F-PLC) in den Eigenschaften der NCU aktivieren.

Hinweis**Auswirkungen auf Telegrammkonfiguration**

Der verwendete Safety Integrated-Modus hat Auswirkungen auf die Telegrammkonfiguration, da im Modus Safety Integrated plus (F-PLC) andere Telegramme verwendet werden, als bei inaktivem Safety Integrated-Modus.

Hinzugefügte oder geänderte Telegramme bleiben jedoch erhalten, sofern diese mit dem neu gewählten Safety Integrated-Modus kompatibel sind.

Stellen Sie ggf. nach der Modusänderung in der Telegrammkonfiguration sicher, dass etwaige Anpassungen noch vorhanden sind.

Sie aktivieren und deaktivieren Safety Integrated (F-PLC) mit dem Service `SafetyModeProvider`.

Hinweis

Wenn Sie Safety Integrated (F-PLC) aktivieren oder deaktivieren, muss sich die PLC im Offline-Modus befinden.

Hinweis

`SafetyModeProvider` ist im Namespace `Siemens.Engineering.MC.Sinumerik` enthalten.

Das folgende Beispiel zeigt, wie Sie den Service `SafetyModeProvider` aufrufen:

SafetyModeProvider aufrufen

```
...
Siemens.Engineering.HW.Device cnc = ...;
try
{
    SafetyModeProvider provider = cnc.GetService<SafetyModeProvider>();
    //Perform the safety mode change:
    provider.SetSafetyMode(SafetyMode.DbSI);
}
catch( (EngineeringException ex) )
{
    // Handle safety mode change failure
}
```

Das folgende Beispiel zeigt, wie Sie die aktuelle Safety Integrated-Einstellung auf dem Gerät abrufen:

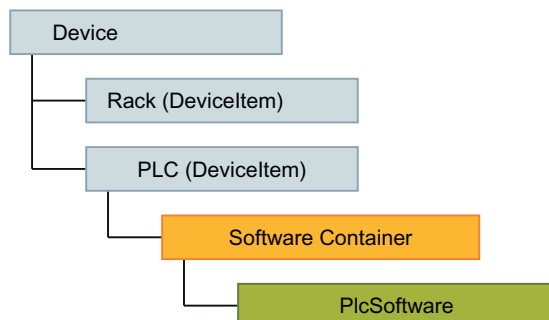
Safety-Einstellung des Geräts aufrufen

```
...
Siemens.Engineering.HW.Device cnc = ...;
try
{
    SafetyModeProvider provider = cnc.GetService<SafetyModeProvider>();
    //Query the safety mode:
    SafetyMode safetyMode = provider.CurrentMode;
}
catch( (EngineeringException ex) )
{
    // Handle any failure
}
```

5.20.5.7 Auf PLC-Software zugreifen

Übersicht

Das folgende Diagramm beschreibt die Darstellung des Software-Containers und der PLC-Software im Objektmodell (Seite 1091):



Das folgende Code-Beispiel zeigt, wie Sie auf eine PLC, unabhängig von konkreter Realisierung (integrierte SINUMERIK PLC, SIMATIC PLC, Software PLC im PC) anhand der Eigenschaft "CPU" zugreifen:

PLCs finden

```
Device ncuDevice = ...
DeviceItem plc = GetPlc(ncuDevice.DeviceItems);

...
DeviceItem GetPlc(DeviceItemComposition deviceItems)
{
    if (deviceItems.Count == 0)
    {
        return null;
    }
    foreach (var deviceItem in deviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.CPU)
            return deviceItem;

        DeviceItem plc = GetPlc(deviceItem.DeviceItems);
        if (plc != null)
            return plc;
    }

    return null;
}
```

Für den konkreten Zugriff auf den PLC-Software-Container lesen Sie weiter unter "Auf Software-Ziel zugreifen".

5.21 Funktionen für SINUMERIK ONE

5.21.1 Einführung

Über TIA Portal-Openness automatisieren Sie das Engineering und steuern das TIA Portal über ein von Ihnen selbst erstelltes Programm.

Für dieses selbst erstellte Programm finden Sie in dieser Hilfe viele Informationen und Beispielcodes. Auch für die TIA Portal-Anwendung "SINUMERIK" können Sie eigene Programme zusammenstellen und anwenden.

Weitere Informationen

Bevor Sie für SINUMERIK aus den nachfolgend gelisteten Beispielcodes ein eigenes Programm zusammenstellen, beachten Sie bitte die übergreifenden Informationen zu Openness, die Sie unter folgenden Stichworten in dieser Hilfe finden:

- Voraussetzungen für TIA Portal Openness
- TIA Portal Openness installieren

- Auf das TIA Portal zugreifen
- TIA Portal Openness-Objektmodell
- Programmierschritte

5.21.2 Typkennung-Kennzeichnung der Komponenten

Jede SINUMERIK-Komponente besitzt eine eindeutige Nummer, die als Typkennung (Typidentifizier) bezeichnet wird. Im Openness-Programmcode können Sie die Typkennung nutzen, um eine Komponente eindeutig zu identifizieren und zu benennen. Beispielsweise lautet die Typkennung für SINUMERIK ONE NCU 1750 "Artikel-Nr.: 6FC5 317-5AA00-0Axx/Vy.z".

Die Typkennung sehen Sie beim Anlegen Ihres Geräts im Dialog "Neues Gerät hinzufügen" sowie in der tabellarischen Geräteübersicht.

Sie können die Typkennung in Ihre Openness-Anwendung kopieren.

Die Typkennung wird in Openness als Aktualparameter beim Aufruf einer Methode, z. B. der `CreateWithItem()`-Methode, erwartet.

Anzeige des Typidentifiers in SINUMERIK aktivieren

1. Wählen Sie in der Projektansicht das Menü "Extras > Einstellungen".
Der Konfigurationsbereich "Einstellungen" wird geöffnet.
2. Wählen Sie in der Sekundärnavigation den Eintrag "Hardware-Konfiguration".
3. Aktivieren Sie die Option "Anzeige des Type Identifiers für Geräte und Module aktivieren".
Die Anzeige des Typidentifiers ist nun aktiv.

5.21.3 Grundlagen

Folgende SINUMERIK-Geräte können Sie im TIA Portal mit Openness anlegen:

- NCU
- NX-Baugruppe
- PPU

Zum Anlegen der SINUMERIK-Geräte verwenden Sie die Methode `CreateWithItem()` der `Devices Collection`.

Hinweis

Alle integrierten Subkomponenten einer SINUMERIK-NCU wie PLC, NCK, CP, HMI und SINAMICS Integrated, werden unterlagert auf der gleichen Ebene automatisch angelegt.

Besonderheiten beim Anlegen der SINUMERIK-Geräte

Die Namen der Baugruppenträger entsprechen den NCU-Typen und sind in der Oberfläche schreibgeschützt. Die Bezeichnungen der Baugruppenträger müssen Sie auch in TIA Portal Openness verwenden. Verwenden Sie die folgenden Standardnamen der Baugruppenträger für die Erstellung eines Geräts über eine Geräteelement-Typkennung mit der `Device CreateWithItem`-Methode:

SINUMERIK-Gerät	Baugruppenträger-Standardnamen
SINUMERIK ONE NCU 1740	NCU 1740
SINUMERIK ONE NCU 1750	NCU 1750
SINUMERIK ONE NCU 1760	NCU 1760
SINUMERIK ONE PPU 1740	PPU 1740

Hinweis

Alternativ können Sie die Parameternamen weglassen. Wenn der Name "null" oder "String.Empty" ist, wird der Standardname verwendet.

Folgende Parameter können mit der Methode `CreateWithItem()` verwendet werden:

- `default name`, z. B. `project.Devices.CreateWithItem("OrderNumber:6FC5 317-5AA00-0AA0/V6.15", "NCU 1750", "TestDevice");`
- `null`, z. B. `project.Devices.CreateWithItem("OrderNumber:6FC5 317-5AA00-0AA0/V6.15", null, "TestDevice");`
- `string.Empty`, z. B. `project.Devices.CreateWithItem("OrderNumber:6FC5 317-5AA00-0AA0/V6.13", string.Empty, "TestDevice");`

Weitere Informationen zu Aufrufparametern der Methode `CreateWithItem()` finden Sie im Kapitel "Erstellen eines Geräts".

Die folgende Tabelle zeigt die Zuordnung der Geräte und ihrer Typkennungen:

SINUMERIK-Gerät	Typkennung
SINUMERIK ONE NCU 1740	Artikel-Nr.: 6FC5 317-4AA00-0Axx/Vy.z
SINUMERIK ONE NCU 1750	Artikel-Nr.: 6FC5 317-5AA00-0Axx/Vy.z
SINUMERIK ONE NCU 1760	Artikel-Nr.: 6FC5 317-6AA00-0Axx/Vy.z
SINUMERIK ONE PPU 1740	Artikel-Nr.: 6FC5 317-4AA00-1xxx/Vy.z

Beim Anlegen der SINUMERIK-Geräte sind Platzhalter in der Typkennung zulässig. Die Platzhalter tauschen Sie später gegen passende gerätespezifische Zeichen aus.

Klassifikation des Geräteelements

Jedes Gerät oder Geräteelement besitzt obligatorische Attribute, die gelesen und geschrieben werden. Weitere Informationen zu den Attributen, die in Openness unterstützt werden, finden Sie im Kapitel "Funktionen auf Geräteelementen".

Die Klassifikationsattribute des Geräteelements sind schreibgeschützt und in der Benutzeroberfläche des TIA Portals nicht sichtbar.

Die Klassifikationsattribute haben den Wert "DeviceItemClassification":

Wert des Klassifikationsattributs	Beschreibung
DeviceItemClassifications.None (0)	Keine Klassifikation.
DeviceItemClassifications.CPU (1)	Das Geräteelement ist eine CPU.
DeviceItemClassifications.HM (2)	Das Geräteelement ist ein Kopfmodul.

Wenn Sie den Wert des Geräteelements über die integrierte PLC abfragen, ist es für ein SINUMERIK-Gerät der Wert "CPU (1)".

Als Kopfmodul fungieren im Openness-Objektmodell die folgenden Komponenten:

- SINAMICS Integrated
- NX-Baugruppe

In allen anderen Fällen ist der Wert des Klassifikationsattributs "DeviceItemClassification" "None" (0).

Geräteelemente über die Eigenschaft "Kopfmodul" finden

Das folgende Beispiel zeigt, wie Sie die Geräteelemente mit der Eigenschaft "Kopfmodul" finden, bevor Sie z. B. ein Telegramm konfigurieren.

Sie können diese Eigenschaft nur lesen, aber nicht an Geräteelemente setzen.

Sinamics Integrated / NX über die Eigenschaft "Kopfmodul" finden

```
foreach (Device device in project.Devices)
{
    foreach (DeviceItem deviceItem in device.DeviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.HM)
        {
            var driveObjectContainer = deviceItem.GetService<DriveObjectContainer>();
            // do something
        }
    }
}
```

Siehe auch

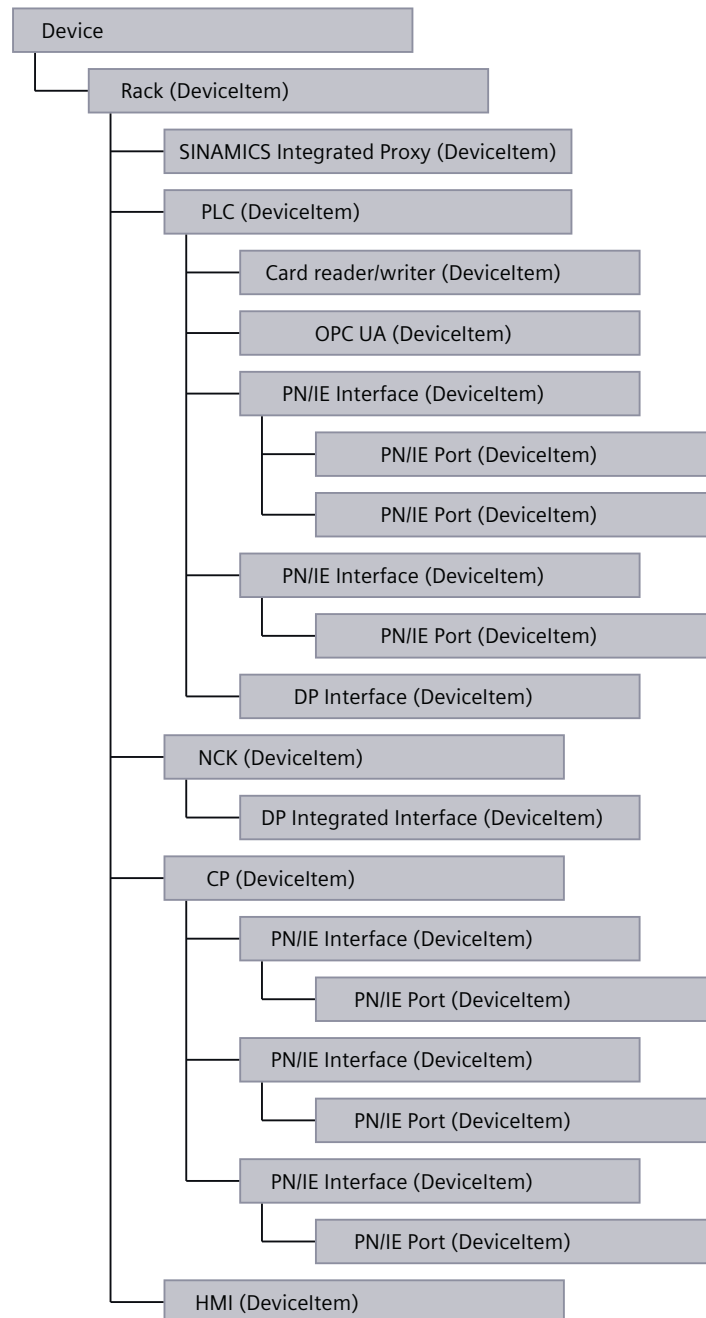
NCU anlegen (Seite 1111)

NX-Baugruppe anlegen (Seite 1111)

5.21.4 Objektmodell

Übersicht

Das folgende Diagramm beschreibt die Objekte, die sich bei SINUMERIK ONE unter "Device" befinden:



Weitere Informationen zu TIA Portal Openness-Objektmodell, finden Sie unter "TIA Portal Openness API".

Verfügbare Namespaces für Telegrammkonfiguration

In SINUMERIK Openness sind für die Telegrammkonfigurationsschnittstellen die folgenden Namespaces gültig:

Namespace	Assembly
Siemens.Engineering.MC.DriveConfiguration (Seite 1093)	Siemens.Engineering.MC.DriveConfiguration in Siemens.Engineering.dll
Siemens.Engineering.MC.Drives (Seite 1102)	Siemens.Engineering.MC.Drives in Siemens.Engineering.dll

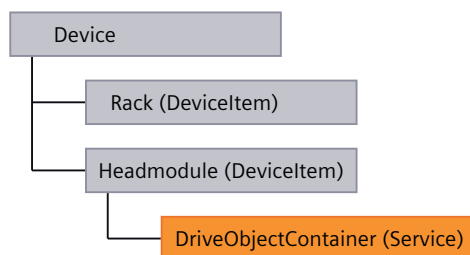
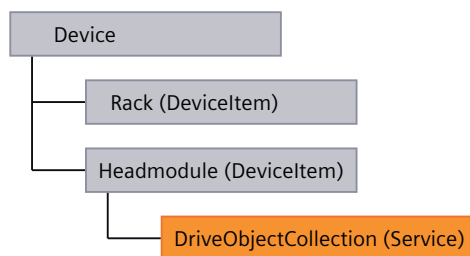
Hinweis

Mit dem Namespace `Siemens.Engineering.MC.DriveConfiguration` stehen Ihnen zusätzliche Funktionen zur Verfügung, z. B. können Sie Antriebsobjekte anlegen, löschen oder ihre Reihenfolge ändern.

Langfristig bleibt der Namespace `Siemens.Engineering.MC.DriveConfiguration` bestehen. Jedoch wird der Namespace `Siemens.Engineering.MC.Drives` aus Kompatibilitätsgründen in den nächsten Openness-Versionen weiterhin unterstützt.

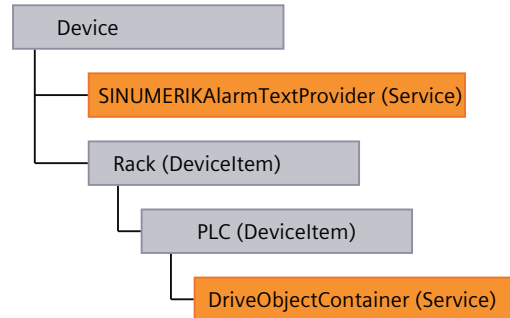
Die folgenden Diagramme beschreiben die Objekte, die sich bei SINUMERIK NX 10.3/15.3 unter "Device" befinden (Verfügbarkeit der Services ist abhängig vom Namespace).

Verwenden Sie für neue Anwendungen den Namespace `Siemens.Engineering.MC.DriveConfiguration` (und damit den Service `DriveObjectCollection`), da der Namespace `Siemens.Engineering.MC.Drives` (und damit der Service `DriveObjectContainer`) zwar aus Kompatibilitätsgründen weiter unterstützt wird, aber weniger Funktionalität anbietet.



Import der Alarmtexte

Das folgende Diagramm bildet das Objekt SinumerikAlarmTextProvider ab, mit dem Alarmtexte **für DB2-Alarme** importiert werden können.



In der Projektnavigation finden Sie die importierten Alarme an den folgenden Stellen:

- in den PLC-Textlisten, z. B. unter "CNC_1 > PLC_1 > PLC-Textlisten"
- in den Projekttexten, z. B. unter "Projekt_1 > Gemeinsame Daten > Projekttexte".

Weitere Informationen und Code-Beispiele finden Sie unter SINUMERIK PLC-Alarmtexte importieren (Seite 1118).

Informationen zum Import der SINUMERIK User Alarms (in Anwendertextlisten) finden Sie in der TIA Portal-Hilfe unter Export/Import von PLC-Meldetextlisten.

5.21.5 Referenz

5.21.5.1 Namespace Siemens.Engineering.MC.DriveConfiguration

DriveObject

DriveObject

Die `DriveObject` Klasse ermöglicht den Zugriff auf das Antriebsobjekt. Über das Antriebsobjekt ist z. B. der Zugriff auf das Telegramm möglich.

Namespace: `Siemens.Engineering.MC.DriveConfiguration`

Assembly: `Siemens.Engineering.MC.DriveConfiguration`
in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
Telegrams	TelegramComposition	read	-	Gibt eine Liste mit den verfügbaren Telegrammen des Antriebsobjekts zurück. Die Liste kann mit der TelegramComposition Klasse verändert werden.
Parent	IEngineeringObject	read	-	Gibt die Referenz auf die übergeordnete Klasse (DriveObjectCollection und DriveObjectContainer) zurück.
Category		read	-	Gibt die Referenz auf die übergeordnete Klasse (DriveObjectCategory) zurück
Name		read	-	Gibt den Namen des Antriebsobjekts zurück.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
GetAttribute	Greift lesend auf ein Attribut eines Antriebsobjekts zu
GetAttributes	Greift lesend auf alle Attribute eines Antriebsobjekts zu
GetAttributeInfos	Greift lesend auf Informationen zu Attributen eines Antriebsobjekts zu
SetAttribute	Greift schreibend auf ein Attribut eines Antriebsobjekts zu
SetAttributes	Greift schreibend auf alle Attribute eines Antriebsobjekts zu
Delete	Löscht die Antriebsobjekt-Instanz, an der "Delete" aufgerufen wird

DriveObjectCollection

DriveObjectCollection

Die DriveObjectCollection ist ein Service des Antriebsobjekts (DeviceItem) vom aktuellen Gerät (Device).

Namespace: Siemens.Engineering.MC.DriveConfiguration

Assembly: Siemens.Engineering.MC.DriveConfiguration
in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Navigatoren** der DriveObjectCollection:

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
DriveObjects	DriveObjectComposition	read	-	Gibt eine Liste mit den verfügbaren Antriebsobjekten zurück. Die Antriebsobjekte ermöglichen den Zugriff auf Antriebsparameter und Telegramme.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
GetAttribute	Greift lesend auf ein Attribut von DriveObjectCollection zu
GetAttributes	Greift lesend auf alle Attribute von DriveObjectCollection zu
GetAttributeInfos	Greift lesend auf Informationen zu Attributen von DriveObjectCollection zu
SetAttribute	Greift schreibend auf ein Attribut von DriveObjectCollection zu
SetAttributes	Greift schreibend auf alle Attribute von DriveObjectCollection zu

DriveObjectComposition

DriveObjectComposition

Die DriveObjectComposition Klasse ermöglicht den Zugriff auf verfügbare Telegramme eines Antriebobjekts und enthält alle Antriebsobjekte einer NCU oder NX-Baugruppe.

Namespace: Siemens.Engineering.MC.DriveConfiguration

Assembly: Siemens.Engineering.MC.DriveConfiguration
in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
Parent	IEngineeringObject	read	-	Gibt die Referenz auf die übergeordnete Klasse (DriveObjectContainer) zurück.
Count		read	-	
IsReadOnly		read	-	

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
GetEnumerator	Ermöglicht die Iteration über die gegebene Menge der Elemente
Contains	Ermittelt, ob die angegebene Instanz in der Menge der Elemente enthalten ist. TRUE: Der Container enthält die Instanz FALSE: Der Container enthält die Instanz nicht
IndexOf	Gibt den Index in der Menge der Elemente für die angefragte Instanz zurück.
Create	Erstellt eine DriveObjectComposition-Klasse
Find	Sucht eine DriveObjectComposition-Klasse

DriveObjectCategory

DriveObjectCategory

Das Enum `DriveObjectCategory` enthält vordefinierte Kategorien der Antriebsobjekte. Auf `DriveObjectCategory` ist ein lesender Zugriff möglich.

Namespace: `Siemens.Engineering.MC.DriveConfiguration`

Assembly: `Siemens.Engineering.MC.DriveConfiguration`
in `Siemens.Engineering.dll`

Die folgende Tabelle enthält die vordefinierten Kategorien der Antriebsobjekte:

Unterstützte Antriebsobjekte in SINUMERIK	DriveObjectCategory
<code>SinumerikDriveAxis</code>	SERVO, ENC, HLA, TM41
<code>SinumerikControlUnit</code>	CU_I, CU_NX_CX
<code>SinumerikInfeed</code>	A_INF, B_INF, S_INF

TelegramComposition

TelegramComposition

Die `TelegramComposition` Klasse ermöglicht den Zugriff auf die Telegramme eines Antriebsobjekts (`DriveObject` (Seite 1093)). Der Aufbau eines Telegramms kann über die `Telegram` (Seite 1097) Klasse ausgelesen werden.

Änderungen an Telegrammobjekten (z. B. Wechsel des Safety-Telegramms), können dazu führen, dass in der `TelegramComposition` das entsprechende Telegrammobjekt gelöscht und ein neues angelegt wird. In diesem Fall müssen Sie die `TelegramComposition` erneut durchsuchen, um das neue Telegrammobjekt (Seite 1097) nach der Änderung wieder zu finden.

Wenn der jeweilige Antriebsobjekttyp keine Telegramme unterstützt, wird der Wert für `TelegramComposition` leer zurückgegeben.

Namespace: `Siemens.Engineering.MC.DriveConfiguration`

Assembly: `Siemens.Engineering.MC.DriveConfiguration`
in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
<code>Parent</code>	<code>IEngineeringObject</code>	read	-	Gibt die Referenz auf die übergeordnete Klasse (<code>DriveObject</code>) zurück.
<code>Count</code>		read	-	
<code>IsReadOnly</code>		read	-	

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
Create(enum TelegramId Id)	Erstellt ein Telegramm, der Telegrammtyp entspricht der ID.
int IndexOf(TelegramType)	Gibt den Index in der Menge der Elemente für die angefragte Instanz zurück.
bool Contains	Ermittelt, ob die angegebene Instanz in der Menge der Elemente enthalten ist. TRUE: Der Container enthält die Instanz FALSE: Der Container enthält die Instanz nicht
IEnumerator GetEnumerator	Ermöglicht IEnumerator<DriveObject> die Iteration über die gegebene Menge der Elemente

Telegram

Telegram

Die **Telegram** Klasse ermöglicht den Zugriff auf den Aufbau eines Telegramms von einem Antriebsobjekt.

Namespace: Siemens.Engineering.MC.DriveConfiguration

Assembly: Siemens.Engineering.MC.DriveConfiguration
in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriffsart	Zu-griff	Beschreibung
Id	Int32	read	-	Gibt die ID des Telegramms zurück.
Type	enum: TelegramType (Seite 1098)	read	-	Gibt den Typ des Telegramms als Enum TelegramType zurück.
Addresses	AddressComposition (Seite 1100)	read	-	Gibt eine AddressComposition mit Informationen zur Adresse zurück.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
GetSize (AddressloType (Seite 1101))	Gibt die Größe der Eingänge oder Ausgänge des Telegramms zurück.
CanSetSize (AddressloType (Seite 1101), Int32, bool)	Gibt true zurück, wenn die Größe des Telegramms wie parametrisiert geändert werden kann.
SetSize	Greift schreibend auf die Telegrammgröße zu
Delete	Löscht die Telegramm-Instanz, an der "Delete" aufgerufen wird
GetAttribute	Greift lesend auf ein Telegramm-Attribut zu
GetAttributes	Greift lesend auf alle Telegramm-Attribute zu

Name	Beschreibung
GetAttributeInfos	Greift lesend auf Informationen zu Telegramm-Attributen zu
SetAttribute	Greift schreibend auf ein Telegramm-Attribut zu
SetAttributes	Greift schreibend auf alle Telegramm-Attribute zu
Delete	Löscht das Telegramm

Eigenschaften von Safety-Telegrammen

Die folgende Tabelle beschreibt die zusätzlichen **Eigenschaften** der Klasse "Telegram" vom Typ "SafetyTelegram".

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
Failsafe_FSourceAddress	UInt32	read	dynamisch	PROFIsafe-Quell-Adresse
Failsafe_FDestinationAddress	UInt32	read/write	dynamisch	PROFIsafe-Zieladresse
Failsafe_FIODBNumber	UInt32	read/write	dynamisch	Safety DB-Nummer, schreibender Zugriff nur möglich, wenn der Eigenschaft "Failsafe_ManualAssignmentFIODBNumber" der Wert "1" zugewiesen ist
Failsafe_FIODBName	string	read	dynamisch	Name Safety-DB
Failsafe_ManualAssignmentFIODBNumber	bool	read/write	dynamisch	Einstellung für manuelle Zuweisung der Eigenschaft "Failsafe_FIODBNumber"
Failsafe_FMonitoringtime	UInt32	read/write	dynamisch	Safety-Überwachungszeit, schreibender Zugriff nur möglich, wenn die Eigenschaft "Failsafe_ManualAssignmentFMonitoringtime" auf "true" gesetzt ist
Failsafe_ManualAssignmentFMonitoringtime	bool	read/write	dynamisch	Einstellung für manuelle Zuweisung der Eigenschaft "Failsafe_FMonitoringtime"

Der Zugriff auf die zusätzliche Eigenschaften erfolgt über `GetAttribute` und `SetAttribute` z. B. `GetAttribute("Failsafe_FSourceAddress")`. Als Rückgabewert wird ein `UInt32` erwartet.

TelegramType

TelegramType

Das Enum `TelegramType` enthält vordefinierte Telegrammtypen.

Namespace: `Siemens.Engineering.MC.DriveConfiguration`
Assembly: `Siemens.Engineering.MC.DriveConfiguration`
in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
MainTelegram	Haupttelegramm: z. B. Telegramm 136
SupplementaryTelegram	Zusatztelegramm: z. B. Telegramm 701
AdditionalTelegram	Erweiterung: freies Telegramm
SafetyTelegram	Safety-Telegramm: z. B. Telegramm 903
ProcessMonitoringTelegram	Prozess-Überwachungstelegramm, z. B. Telegramm 1100

Hinweis

Das Drehmoment-Telegramm (`TorqueTelegram`) wird nicht unterstützt, obwohl es in der TIA Portal Openness-API im Kontext SINUMERIK verfügbar ist.

TelegramPart

TelegramPart

Das Enum `TelegramPart` enthält die Bestandteile eines Telegramms.

Namespace: `Siemens.Engineering.MC.DriveConfiguration`

Assembly: `Siemens.Engineering.MC.DriveConfiguration`
in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
Unknown	Unbekannter Bestandteil eines Telegramms
Overall	Alle Bestandteile eines Telegramms in einem Telegramm: BaseTelegram, Extension und PKW ¹⁾ eines Telegramms
BaseTelegram	Hauptbestandteil eines Telegramms
Extension	Erweiterung eines Telegramms
PKW	Parameter-Kennung-Wert eines Telegramms

¹⁾ PKW ist im SINUMERIK-Kontext nicht verfügbar.

TelegramId

TelegramId

Das Enum `TelegramId` enthält die Telegrammnummern, die für die Kommunikation zwischen der PLC und dem Antrieb relevant sind. Die IDs sind über den PROFIdrive-Standard festgelegt.

Namespace: `Siemens.Engineering.MC.DriveConfiguration`

Assembly: `Siemens.Engineering.MC.DriveConfiguration`
in `Siemens.Engineering.dll`

AddressComposition

AddressComposition

Die AddressComposition Klasse repräsentiert die Adresse eines Telegramms.

namespace: Siemens.Engineering.MC.DriveConfiguration

Assembly: Siemens.Engineering.MC.DriveConfiguration
in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zu- griffsart	Zugriff	Beschreibung
IoType	enum: AddressIoType (Seite 1101)	read	-	Gibt Informationen zum Typ der Adresse zurück.
Context	enum: AddressContext (Seite 1101)	read	Zugriff nur über GetAttribute oder SetAttribute möglich	Gibt Informationen zum Kontext der Adresse zurück.
StartAddresses	Int32	read/ write	-	Gibt die Startadresse des Telegramms zurück oder legt die Startadresse fest.
Length	Int32	read	-	Gibt die Länge des Telegramms zurück.
IndexOf	int32	read	-	Gibt den Index in der Menge der Elemente für die angefragte Instanz zurück.
Contains	bool	read	-	Ermittelt, ob die angegebene Instanz in der Menge der Elemente enthalten ist. TRUE: Der Container enthält die Instanz FALSE: Der Container enthält die Instanz nicht
GetEnumerator	IEnumerator<DriveObject>	read	-	Ermöglicht die Iteration über die gegebene Menge der Elemente

Hinweis

Wenn Sie über Address.Parent zurück zum Telegramm navigieren möchten, wird statt des Namespace Siemens.Engineering.MC.Drives.Telegram der Namespace Siemens.Engineering.MC.DriveConfiguration.Telegram aufgerufen.

Weitere Informationen zu TIA Portal Openness-Bibliotheken, finden Sie unter "Standard-Bibliotheken".

AddressIoType

AddressIoType

Das Enum AddressIoType enthält Informationen zum Typ der Adresse.

Namespace: Siemens.Engineering.MC.DriveConfiguration

Assembly: Siemens.Engineering.MC.DriveConfiguration
in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
AddressIoType.None	Der IO-Typ ist nicht anwendbar
AddressIoType.Input	Typ ist eine Eingangsadresse
AddressIoType.Output	Typ ist eine Ausgangsadresse
AddressIoType.Diagnosis	Typ ist eine Diagnoseadresse
AddressIoType.Substitute	Typ ist eine Ersatzadresse

Weitere Informationen zu TIA Portal Openness-Bibliotheken, finden Sie unter "Standard-Bibliotheken".

AddressContext

AddressContext

Das Enum AddressContext enthält Informationen zum Kontext der Adresse.

Namespace: Siemens.Engineering.MC.DriveConfiguration

Assembly: Siemens.Engineering.MC.DriveConfiguration
in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
AddressContext.None	Für die Adresse wurde kein Kontext gefunden
AddressContext.Device	Kontext ist eine Geräteadresse
AddressContext.Head	Kontext ist eine Kopfadresse

Weitere Informationen zu TIA Portal Openness-Bibliotheken, finden Sie unter "Standard-Bibliotheken".

5.21.5.2 Namespace Siemens.Engineering.MC.Drives

DriveObject

DriveObject

Die `DriveObject` Klasse ermöglicht den Zugriff auf das Antriebsobjekt. Über das Antriebsobjekt ist z. B. der Zugriff auf das Telegramm möglich.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
Telegramms	TelegramComposition	read	-	Gibt eine Liste mit den verfügbaren Telegrammen des Antriebsobjekts zurück. Die Liste kann mit der <code>TelegramComposition</code> Klasse verändert werden.
Parent	IEngineeringObject	read	-	Gibt die Referenz auf die übergeordnete Klasse (<code>DriveObjectContainer</code>) zurück.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
<code>GetAttribute</code>	Greift lesend auf ein Attribut eines Antriebsobjekts zu
<code>SetAttribute</code>	Greift schreibend auf ein Attribut eines Antriebsobjekts zu
<code>GetEnumerator</code>	Ermöglicht die Iteration über die gegebene Menge der Elemente

Siehe auch

[TelegramComposition](#) (Seite 1103)

DriveObjectContainer

DriveObjectContainer

Der `DriveObjectContainer` ist ein Service des Antriebsobjekts (`DeviceItem`) vom aktuellen Gerät (`Device`).

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Navigatoren** des `DriveObjectContainer`:

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
<code>DriveObjects</code>	<code>DriveObjectComposition</code>	read	-	Gibt eine Liste mit den verfügbaren Antriebsobjekten zurück. Die Antriebsobjekte ermöglichen den Zugriff auf Antriebsparameter und Telegramme.
<code>Parent</code>	<code>IEngineeringObject</code>	read	-	Gibt die Referenz auf die übergeordnete Klasse (<code>DeviceItem</code>) zurück.

DriveObjectComposition

DriveObjectComposition

Die `DriveObjectComposition` Klasse ermöglicht den Zugriff auf verfügbare Telegramme eines Antriebsobjekts.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
<code>IndexOf</code>	<code>int32</code>	read	-	Gibt den Index in der Menge der Elemente für die angefragte Instanz zurück.
<code>Contains</code>	<code>bool</code>	read	-	Ermittelt, ob die angegebene Instanz in der Menge der Elemente enthalten ist. <code>TRUE</code> : Der Container enthält die Instanz <code>FALSE</code> : Der Container enthält die Instanz nicht
<code>GetEnumerator</code>	<code>IEnumerator<DriveObject></code>	read	-	Ermöglicht die Iteration über die gegebene Menge der Elemente

TelegramComposition

TelegramComposition

Die `TelegramComposition` Klasse ermöglicht den Zugriff auf die Telegramme eines Antriebsobjekts (`DriveObject` (Seite 1102)). Der Aufbau eines Telegramms kann über die `Telegram` (Seite 1105) Klasse ausgelesen werden.

Änderungen an Telegrammobjekten (z. B. Wechsel des Safety-Telegrams), können dazu führen, dass in der `TelegramComposition` das entsprechende Telegrammobjekt gelöscht

und ein neues angelegt wird. In diesem Fall müssen Sie die `TelegramComposition` erneut durchsuchen, um das neue Telegrammobjekt (Seite 1105) nach der Änderung wieder zu finden.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
<code>CanInsertAdditionalTelegram(Int32, Int32)</code>	Gibt <code>true</code> zurück, wenn eine Erweiterung entsprechend den parametrisierten Größen (Eingangs- und Ausgangsgröße) erstellt werden kann.
<code>InsertAdditionalTelegram(Int32, Int32)</code>	Erstellt für das Antriebsobjekt eine Erweiterung entsprechend den parametrisierten Größen und gibt <code>true</code> zurück, wenn die Erweiterung eingefügt werden konnte. Im Fehlerfall wird eine <code>EngineeringTargetInvocationException</code> ausgelöst.
<code>CanInsertSupplementaryTelegram(Int32)</code>	Gibt <code>true</code> zurück, wenn ein Zusatztelegramm entsprechend der parametrisierten Telegrammnummer erstellt werden kann.
<code>InsertSupplementaryTelegram(Int32)</code>	Erstellt das Zusatztelegramm mit der parametrisierten Telegrammnummer und gibt <code>true</code> zurück, wenn das Telegramm eingefügt werden konnte. Im Fehlerfall wird eine <code>EngineeringTargetInvocationException</code> ausgelöst.
<code>CanInsertSafetyTelegram(Int32)</code>	Gibt <code>true</code> zurück, wenn ein Safety-Telegramm entsprechend der parametrisierten Telegrammnummer erstellt werden kann.
<code>InsertSafetyTelegram(Int32)</code>	Erstellt das Zusatztelegramm mit der parametrisierten Telegrammnummer und gibt <code>true</code> zurück, wenn das Telegramm eingefügt werden konnte. Im Fehlerfall wird eine <code>EngineeringTargetInvocationException</code> ausgelöst.
<code>EraseTelegram(TelegramType)</code>	Gibt <code>true</code> zurück, wenn das parametrisierte Telegramm gelöscht werden konnte. Standardtelegramme können nicht gelöscht werden. Im Fehlerfall wird eine <code>EngineeringTargetInvocationException</code> ausgelöst.

Name	Beschreibung
<code>Find(TelegramType)</code>	Gibt das Objekt Telegram (Seite 1105) zurück, wenn es über den parametrisierten Telegrammtyp gefunden werden konnte. null, wenn das Telegramm nicht gefunden wird. Beispiel <code>Telegram telegram = telegrams.Find(TelegramType.MainTelegram);</code>
<code>int IndexOf(TelegramType)</code>	Gibt den Index in der Menge der Elemente für die angefragte Instanz zurück.
<code>Parent</code>	Gibt die Referenz auf die übergeordnete Klasse (DriveObject) zurück.
<code>bool Contains</code>	Ermittelt, ob die angegebene Instanz in der Menge der Elemente enthalten ist. TRUE: Der Container enthält die Instanz FALSE: Der Container enthält die Instanz nicht
<code>IEnumerator GetEnumerator</code>	Ermöglicht <code>IEnumerator<DriveObject></code> die Iteration über die gegebene Menge der Elemente

Telegram

Telegram

Die `Telegram` Klasse ermöglicht den Zugriff auf den Aufbau eines Telegramms von einem Antriebsobjekt.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriffsart	Zu-griff	Beschreibung
<code>TelegramNumber</code>	<code>Int32</code>	read	-	Gibt die Nummer des Haupttelegramms zurück oder legt die Nummer fest. Ein freies Telegramm hat die Nummer 999.
<code>Type</code>	<code>enum:TelegramType</code> (Seite 1107)	read	-	Gibt den Typ des Telegramms als Enum <code>TelegramType</code> zurück.
<code>Addresses</code>	<code>AddressComposition</code> (Seite 1108)	read	-	Gibt eine <code>AddressComposition</code> mit Informationen zur Adresse zurück.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
CanChangeTelegram(Int32)	Gibt <code>true</code> zurück, wenn das Telegramm in den parametrisierten Standardtyp geändert werden kann.
GetSize(AddressloType (Seite 1109))	Gibt die Größe der Eingänge oder Ausgänge des Telegramms zurück.
CanChangeSize(AddressloType (Seite 1109), Int32, bool)	Gibt <code>true</code> zurück, wenn die Größe des Telegramms wie parametrisiert geändert werden kann. Standardtelegramme können nur vergrößert werden. Das Beibehalten der bisherigen Telegrammadresse wird berücksichtigt, wenn die Option mit <code>true</code> parametrisiert ist.
ChangeSize(AddressloType (Seite 1109), Int32, bool)	Gibt <code>true</code> zurück, wenn die Größe des Telegramms wie parametrisiert geändert werden konnte. Das Beibehalten der bisherigen Telegrammadresse wird berücksichtigt, wenn die Option mit <code>true</code> parametrisiert ist.
GetEnumerator	Ermöglicht die Iteration über die gegebene Menge der Elemente

Eigenschaften von Safety-Telegrammen

Die folgende Tabelle beschreibt die zusätzlichen **Eigenschaften** der Klasse "Telegram" vom Typ "SafetyTelegram".

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
Failsafe_FSourceAddress	UInt32	read	Zugriff nur über <code>GetAttribute</code> oder <code>SetAttribute</code> möglich	PROFIsafe-Quell-Adresse
Failsafe_FDestinationAddress	UInt32	read/write	Zugriff nur über <code>GetAttribute</code> oder <code>SetAttribute</code> möglich	PROFIsafe-Zieladresse
Failsafe_FIODBNumber	UInt32	read/write	Zugriff nur über <code>GetAttribute</code> oder <code>SetAttribute</code> möglich	Safety DB-Nummer, schreibender Zugriff nur möglich, wenn der Eigenschaft "Failsafe_Manual AssignmentFIODBNumber" der Wert "1" zugewiesen ist

Name	Datentyp	Zugriffsart	Zugriff	Beschreibung
Failsafe_FIODBName	string	read	Zugriff nur über GetAttribute oder SetAttribute möglich	Name Safety-DB
Failsafe_ManualAssignmentFIODBNumber	bool	read/write	Zugriff nur über GetAttribute oder SetAttribute möglich	Einstellung für manuelle Zuweisung der Eigenschaft "Failsafe_FIODBNumber"
Failsafe_FMonitoringtime	UInt32	read/write	Zugriff nur über GetAttribute oder SetAttribute möglich	Safety-Überwachungszeit, schreibender Zugriff nur möglich, wenn die Eigenschaft "Failsafe_ManualAssignmentFMonitoringtime auf "true" gesetzt ist
Failsafe_ManualAssignmentFMonitoringtime	bool	read/write	Zugriff nur über GetAttribute oder SetAttribute möglich	Einstellung für manuelle Zuweisung der Eigenschaft "Failsafe_FMonitoringtime"

Der Zugriff auf die zusätzlichen Eigenschaften erfolgt über `GetAttribute` und `SetAttribute` z. B. `GetAttribute("Failsafe_FSourceAddress")`. Als Rückgabewert wird ein `UInt32` erwartet.

TelegramType

TelegramType

Das Enum `TelegramType` enthält vordefinierte Telegrammtypen.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
MainTelegram	Haupttelegramm: z. B. Telegramm 136
SupplementaryTelegram	Zusatztelegramm: z. B. Telegramm 701
AdditionalTelegram	Erweiterung: freies Telegramm
SafetyTelegram	Safety-Telegramm: z. B. Telegramm 903

Hinweis

Das Drehmoment-Telegramm (TorqueTelegram) wird nicht unterstützt, obwohl es in der TIA Portal Openness-API im Kontext SINUMERIK verfügbar ist.

AddressComposition**AddressComposition**

Die AddressComposition Klasse repräsentiert die Adresse eines Telegramms.

Namespace: Siemens.Engineering.MC.Drives

Assembly: Siemens.Engineering.MC.Drives in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zu-griffsart	Zugriff	Beschreibung
IoType	enum: AddressIoType (Seite 1109)	read	-	Gibt Informationen zum Typ der Adresse zurück.
Context	enum: AddressContext (Seite 1109)	read	Zugriff nur über GetAttribute oder SetAttribute möglich	Gibt Informationen zum Kontext der Adresse zurück.
StartAddresses	Int32	read/write	-	Gibt die Startadresse des Telegramms zurück oder legt die Startadresse fest.
Length	Int32	read	-	Gibt die Länge des Telegramms zurück.
IndexOf	int32	read	-	Gibt den Index in der Menge der Elemente für die angefragte Instanz zurück.
Contains	bool	read	-	Ermittelt, ob die angegebene Instanz in der Menge der Elemente enthalten ist. TRUE: Der Container enthält die Instanz FALSE: Der Container enthält die Instanz nicht
GetEnumerator	IEnumerator<DriveObject>	read	-	Ermöglicht die Iteration über die gegebene Menge der Elemente

Hinweis

Wenn Sie über `Address.Parent` zurück zum Telegramm navigieren möchten, wird statt des Namespace `Siemens.Engineering.MC.Drives.Telegram` der Namespace `Siemens.Engineering.MC.DriveConfiguration.Telegram` aufgerufen.

Weitere Informationen zu TIA Portal Openness-Bibliotheken, finden Sie unter "Standard-Bibliotheken".

AddressContext**AddressContext**

Das Enum `AddressContext` enthält Informationen zum Kontext der Adresse.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
<code>AddressContext.None</code>	Für die Adresse wurde kein Kontext gefunden
<code>AddressContext.Device</code>	Kontext ist eine Geräteadresse
<code>AddressContext.Head</code>	Kontext ist eine Kopfadresse

Weitere Informationen zu TIA Portal Openness-Bibliotheken, finden Sie unter "Standard-Bibliotheken".

AddressIoType**AddressIoType**

Das Enum `AddressIoType` enthält Informationen zum Typ der Adresse.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
<code>AddressIoType.None</code>	Der IO-Typ ist nicht anwendbar
<code>AddressIoType.Input</code>	Typ ist eine Eingangsadresse
<code>AddressIoType.Output</code>	Typ ist eine Ausgangsadresse
<code>AddressIoType.Diagnosis</code>	Typ ist eine Diagnoseadresse
<code>AddressIoType.Substitute</code>	Typ ist eine Ersatzadresse

Weitere Informationen zu TIA Portal Openness-Bibliotheken, finden Sie unter "Standard-Bibliotheken".

5.21.5.3 ArchiveProvider

ArchiveProvider

Die Klasse ArchiveProvider dient zum Generieren der PLC-Archive.

Sie können auch Archive über Openness erzeugen, wenn Safety Integrated-Modus aktiviert ist.

Namespace: Siemens.Engineering.HW.Utilities

Assembly: Siemens.Engineering.HW.Utilities
in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die Methoden der Klasse ArchiveProvider:

Name	Parameter	Beschreibung
Archive	DeviceItem plc FileInfo path Siemens.Engineering.MC.Sinumerik.SinumerikArchiveMode String comment (optional) String author (optional)	Erstellt ein PLC-Archiv
Retrieve	FileInfo archivePath	Konsistentes Laden des Archivs ins TIA Portal

Hinweis

Konsistentes Laden von der F-CPU

Achten Sie vor dem Erstellen eines Archivs darauf, dass Sie im Safety Administration Editor unter "Einstellungen > Erweiterte Einstellungen" die Option "Konsistentes Laden von der F-CPU" setzen, damit Sie auch Geräte mit aktiviertem Safety-Integrated Modus als neue Station in das TIA Portal-Projekt laden können.

(Funktion "Werkzeuge > Gerät aus SINUMERIK-Archiv hochladen > Gerät als neue Station hochladen (Hardware und Software)"

5.21.6 Codebeispiel

5.21.6.1 Allgemein

Die folgenden Codebeispiele beschreiben die grundsätzliche Vorgehensweise für verschiedene Anwendungsfälle. Der Code ist nicht zwingend vollständig und kompilierbar.

5.21.6.2 Erste Schritte in SINUMERIK durchführen

- Verbinden Sie die TIA Portal Openness-Anwendung mit dem TIA Portal.
- Öffnen Sie Ihr Projekt.

Das folgende Beispiel zeigt, wie Sie ermitteln können, welche SINUMERIK Toolbox-Version installiert ist.

SINUMERIK Toolbox-Version ermitteln

```
using Siemens.Engineering;

if (tiaProcess.InstalledSoftware.Any(sw => sw.Name.Equals("SINUMERIK Toolbox") &&
sw.Version.Equals("V17")))
{
    Console.WriteLine("SINUMERIK Toolbox is available");
}
// "V17" is the current SINUMERIK version started at October 2021.
```

5.21.6.3 NCU anlegen

Mit der Methode `CreateWithItem()` der `Devices` Collection legen Sie eine SINUMERIK-NCU an. Die SINUMERIK-NCUs werden über Parameter der Methode spezifiziert. Das Format der Parameter ist nachfolgend beschrieben.

NCU anlegen

Format der Parameter für SINUMERIK-NCU:

```
CreateWithItem(@"OrderNumber:mlfb/FirmwareVersion/",
"StandardSubrackName", "NameOfTheDevice")
```

Das folgende Beispiel zeigt, wie Sie eine Steuerung SINUMERIK ONE NCU 1750 anlegen.

SINUMERIK ONE NCU 1750 anlegen

```
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
Project tiaproject= portal.Projects.Open("..."); //The path of the project

Device NCUDevice = tiaproject.Devices.CreateWithItem("@OrderNumber:6FC5 317-5AA00-0AA0/
V6.15/", "NCU 1750", "TestDevice");
```

5.21.6.4 NX-Baugruppe anlegen

Mit der `Device` `CreateWithItem`-Methode legen Sie eine NX-Baugruppe an. Anschließend verbinden Sie die NX-Baugruppe mit einer NCU.

Folgende Typkennungen werden für SINUMERIK NX-Baugruppen verwendet:

SINUMERIK NX-Baugruppe	Typkennung
SINUMERIK NX10.3	OrderNumber:6SL3 040-1NC00-0Axx/Vy.z
SINUMERIK NX15.3	OrderNumber:6SL3 040-1NB00-0Axx/Vy.z

Das folgende Beispiel zeigt, wie Sie eine SINUMERIK NX-Baugruppe anlegen.

NX-Baugruppe anlegen

```
project.Devices.CreateWithItem("OrderNumber:6SL3040-1NC00-0AA0/
V5.2", "MyNXDevice", "TestDevice");
```

Versionskompatibilität

Die Firmware-Version der NX-Baugruppe muss mit der Firmware-Version von SINAMICS Integrated übereinstimmen und mit der Firmware-Version der NCU kompatibel sein.

Die von SINAMICS Integrated abweichende NX Firmware-Version kann über Openness nicht zugeordnet werden.

Die folgende Tabelle zeigt die Versionskompatibilität für SINUMERIK ONE:

NCU-Firmware (SINUMERIK ONE)	SINAMICS Integrated/NX-Firmware
V6.15	V5.2
V6.20	
V6.21	
V6.22	

5.21.6.5 NX-Baugruppe mit NCU verbinden

NX-Baugruppe mit einer NCU verbinden

Um eine NX-Baugruppe mit einer NCU über den Subnetztyp "ProfibusIntegrated" zu verbinden, muss der Dienst "NetworkInterface" über das Kopfmodul von NX geladen werden.

Das folgende Beispiel zeigt, wie Sie den Dienst "NetworkInterface" laden.

NetworkInterface-Dienst laden

```
foreach (Device device in project.Devices)
{
    foreach (DeviceItem deviceItem in device.DeviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.HM)
        {
            var networkInterface = deviceItem.GetService<NetworkInterface>();
            // do something
        }
    }
}
```

Das folgende Beispiel zeigt, wie Sie eine NX-Baugruppe über "ProfibusIntegrated" mit einer NCU verbinden:

NX-Baugruppe über ProfibusIntegrated verbinden

```
Subnet pbiSubnet = ...;
Node node = networkInterface.Nodes.FirstOrDefault();
node.ConnectToSubnet(pbiSubnet);
```

Die DP-Adresse wird in Openness über DRIVE-CLiQ Port der fest NX zugeordnet. Jedes Port-Label verfügt über eine feste DP-Integrated-Adresse.

Die DP-Adresse muss vor dem Verbinden mit der NCU der NX zugeordnet werden.

5.21.6.6 Auf NCK-Ereignisse zugreifen

NCK-Ereignisse

Das NCK-Modul ist laut dem SINUMERIK-Objektmodell (Seite 1091) ein `DeviceItem`.

Mit den folgenden Attributen des NCK-Moduls greifen Sie auf die NCK-Ereignisse zu und können die NCK-Ereignisse in TIA Portal-Openness konfigurieren:

Name	Datentyp	Zugriffsart	Beschreibung
<code>HardwareInterruptNckToPlcSignalExchangeActive</code>	Bool	r/w	Aktivieren/deaktivieren des Ereignisses
<code>HardwareInterruptNckToPlcSignalExchangeEventName</code>	String	r/w	Ereignisname
<code>HardwareInterruptNckToPlcSignalExchangeInterrupt</code>	<code>Siemens.Engineering.SW.Blocks.OB</code>	r/w	Der dem Ereignis zugewiesene OB
<code>HardwareInterruptNckToPlcSignalExchangePriority</code>	Int32	r/w	Ereignispriorität

Alle NCK-Ereignisse werden über `HardwareInterrupt` (Prozessalarm-OB) getriggert. Die Prozessalarm-OBs unterbrechen die zyklische Programmbearbeitung aufgrund eines Hardware-Ereignisses.

Das folgende Beispiel zeigt, wie Sie den Ereignisnamen für NCK-Modul ermitteln:

Ereignisnamen für NCK-Modul ermitteln

```
DeviceItem nck = ...;
string eventName =
(string)nck.GetAttribute("HardwareInterruptNckToPlcSignalExchangeEventName");
```

Das folgende Beispiel zeigt, wie Sie den HardwareInterrupt setzen:

HardwareInterrupt setzen

```
... DeviceItem nck = ...;
OB ob40 = ... try
{
    nck.SetAttribute("HardwareInterruptNckToPlcSignalExchangeInterrupt", ob40);
}
catch( (EngineeringException ex) )
{
    // Handle setting failure
}
```

5.21.6.7 Archive erstellen

Archive für Serieninbetriebnahme erstellen

Mit TIA Portal-Openness erstellen und exportieren Sie SINUMERIK-Archive, um z. B. die Serieninbetriebnahme zu vereinfachen.

Sie erstellen SINUMERIK-Archive über die TIA Portal-Projekteigenschaft `HwUtilities` mit dem Service `SinumerikArchiveProvider`.

Das folgende Beispiel zeigt, wie Sie den Service `SinumerikArchiveProvider` aufrufen:

SinumerikArchiveProvider aufrufen

```
Project project = ...;
SinumerikArchiveProvider archiveProvider =
project.HwUtilities.Find("SinumerikArchiveProvider") as SinumerikArchiveProvider;

if (archiveProvider != null)
{
    // Work with the provider
}
```

Das folgende Beispiel zeigt, wie Sie ein PLC-Archiv erstellen, das die Hardware-Informationen und alle Datenbausteine enthält:

PLC-Archiv erstellen

```
...
Siemens.Engineering.HW.DeviceItem plc = ...;

try {
    // The file extension is required
    string archivePath = string.Format(@"D:\some_path\{0}.dsf", plc.Name);

    // Comment and author arguments are optional
    archiveProvider.Archive(plc, new FileInfo(archivePath),
SinumerikArchivationMode.HardwareAndAllProgramBlocks);
}
catch (EngineeringTargetInvocationException ex)
{
    // Handle archive failure
}
```

Sie können die Methode folgenderweise verwenden:

```
void Archive(DeviceItem plc, FileInfo path,
SinumerikArchivationMode content, String comment, String author)
```

Das folgende Beispiel zeigt, wie Sie die Software-Anteile eines bereits erstellten PLC-Archivs aktualisieren:

Archivanteile aktualisieren

```
...
Siemens.Engineering.HW.DeviceItem plc_1 = ...;
Siemens.Engineering.HW.DeviceItem plc_1_copy = ...;
try {
    // The file extension is required
    string archivePath = @"D:\some_path\SinumerikArchive.dsfc";

    // Create a Sinumerik archive with HardwareAndAllProgramBlocks
    archiveProvider.Export(plc_1, new FileInfo(archivePath),
SinumerikArchivationMode.HardwareAndAllProgramBlocks);

    // Update the software part in the previously created archive using
UpdateProgramBlocksOfArchive method
archiveProvider.UpdateProgramBlocksOfArchive(plc_1_copy, new FileInfo(archivePath));
}
catch (EngineeringException ex)
{
    // Handle export failure
}
```

Das folgende Beispiel zeigt, wie Sie die SINUMERIK-Archive mit der F-Adressenzuweisung erstellen:

SINUMERIK-Archive mit F-Adressenzuweisung erstellen

```
...
Siemens.Engineering.HW.DeviceItem plc_1 = ...;
Siemens.Engineering.HW.DeviceItem plc_1_copy = ...;

try{
    // The file extension is required
    string archivePath = @"D:\some_path\SinumerikArchive.dsfc";

    // Create a SINUMERIK archive with F-address assignment using
CreateFAddressAssignmentArchive method

archiveProvider.CreateFAddressAssignmentArchive(plc_1, plc_1_copy, new FileInfo(archivePath),
["Comment", "Author name"]);
}
catch (EngineeringTargetInvocationException ex) {
    // Handle archive failure
}
```

Das folgende Beispiel zeigt, wie Sie ein Archiv ins TIA Portal konsistent laden können (nur PLC-Anteil von SINUMERIK-DSF-Archive vom Typ Setup):

Archiv konsistent ins TIA Portal laden

```
...
try{
    // The path of the
    archivestringarchivePath=@"D:\some_path\SinumerikArchive.dsf";

    // Retrieve an already archived SINUMERIK device

    Device retrievedDevice=archiveProvider.Retrieve(newFileInfo(archivePath));
}
catch(EngineeringTargetInvocationException ex)
{
    // Handle retrieve failure
}
```

5.21.6.8 Safety Integrated aktivieren

Safety Integrated aktivieren

Mit TIA Portal Openness können Sie Safety Integrated (F-PLC) in den Eigenschaften der NCU aktivieren.

Hinweis**Auswirkungen auf Telegrammkonfiguration**

Der verwendete Safety Integrated-Modus hat Auswirkungen auf die Telegrammkonfiguration, da im Modus Safety Integrated plus (F-PLC) andere Telegramme verwendet werden, als bei inaktivem Safety Integrated-Modus.

Hinzugefügte oder geänderte Telegramme bleiben jedoch erhalten, sofern diese mit dem neu gewählten Safety Integrated-Modus kompatibel sind.

Stellen Sie ggf. nach der Modusänderung in der Telegrammkonfiguration sicher, dass etwaige Anpassungen noch vorhanden sind.

Sie aktivieren und deaktivieren Safety Integrated (F-PLC) mit dem Service `SafetyModeProvider`.

Hinweis

Wenn Sie Safety Integrated (F-PLC) aktivieren oder deaktivieren, muss sich die PLC im Offline-Modus befinden.

Hinweis

`SafetyModeProvider` ist im Namespace `Siemens.Engineering.MC.Sinumerik` enthalten.

Das folgende Beispiel zeigt, wie Sie den Service `SafetyModeProvider` aufrufen:

SafetyModeProvider aufrufen

```
...
Siemens.Engineering.HW.Device cnc = ...;
try
{
    SafetyModeProvider provider = cnc.GetService<SafetyModeProvider>();
    //Perform the safety mode change:
    provider.SetSafetyMode(SafetyMode.DbSI);
}
catch( (EngineeringException ex) )
{
    // Handle safety mode change failure
}
```

Das folgende Beispiel zeigt, wie Sie die aktuelle Safety Integrated-Einstellung auf dem Gerät abrufen:

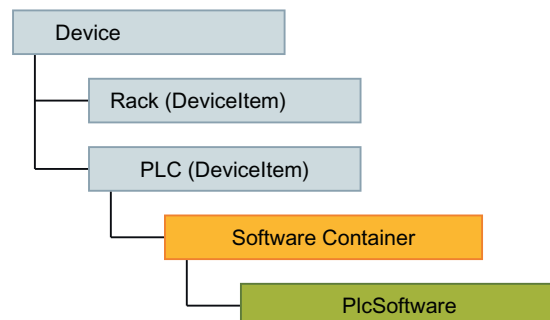
Safety-Einstellung des Geräts aufrufen

```
...
Siemens.Engineering.HW.Device cnc = ...;
try
{
    SafetyModeProvider provider = cnc.GetService<SafetyModeProvider>();
    //Query the safety mode:
    SafetyMode safetyMode = provider.CurrentMode;
}
catch( (EngineeringException ex) )
{
    // Handle any failure
}
```

5.21.6.9 Auf PLC-Software zugreifen

Übersicht

Das folgende Diagramm beschreibt die Darstellung des Software-Containers und der PLC-Software im Objektmodell (Seite 1091):



Das folgende Code-Beispiel zeigt, wie Sie auf eine PLC, unabhängig von konkreter Realisierung (integrierte SINUMERIK PLC, SIMATIC PLC, Software PLC im PC) anhand der Eigenschaft "CPU" zugreifen:

PLCs finden

```
Device ncuDevice = ...
DeviceItem plc = GetPlc(ncuDevice.DeviceItems);

...
DeviceItem GetPlc(DeviceItemComposition deviceItems)
{
    if (deviceItems.Count == 0)
    {
        return null;
    }
    foreach (var deviceItem in deviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.CPU)
            return deviceItem;

        DeviceItem plc = GetPlc(deviceItem.DeviceItems);
        if (plc != null)
            return plc;
    }

    return null;
}
```

Für den konkreten Zugriff auf den PLC-Software-Container lesen Sie weiter unter "Auf Software-Ziel zugreifen".

5.21.6.10 SINUMERIK PLC-Alarmtexte importieren

Die Klasse `SINUMERIKAlarmTextProvider` dient zum Importieren der Texte der SINUMERIK-DB2-Alarme im TS- oder CSV-Format.

Hinweis

`SinumerikAlarmTextProvider` ist im Namespace `Siemens.Engineering.MC.Sinumerik` enthalten.

Sie importieren SINUMERIK PLC-Alarmtexte über Navigator `AlarmTextImporter` mit dem Service `SinumerikAlarmTextProvider`.

Das folgende Beispiel zeigt, wie Sie die Texte der SINUMERIK-DB2-Alarme von einer SINUMERIK NCU über `SinumerikAlarmTextProvider` importieren:

Über `SinumerikAlarmTextProvider` Texte importieren

```
...Siemens.Engineering.HW.Devicencu=...;
System.Collections.IEnumerable<System.IO.FileInfo>inputFiles=...;
try
{
    // Get SinumerikAlarmTextProvider service from a SINUMERIK device
    SinumerikAlarmTextProviderprovider=ncu.GetService<SinumerikAlarmTextProvider>();
    // Call import function through the navigator:
    provider.AlarmTextImporter.ImportAlarmTexts(inputFiles);
}
catch (EngineeringExceptionex)
{
    // Handle import failures
}
```

Informationen zum Import der SINUMERIK User Alarms (in Anwendertextlisten) finden Sie in der TIA Portal-Hilfe unter Export/Import von PLC-Meldetextlisten.

5.21.6.11 Beispiele für Namespace `Siemens.Engineering.MC.DriveConfiguration`

Telegramme vorbereiten

Die Antriebskommunikation einer SINUMERIK-NCU erfolgt mithilfe von Telegrammen über die Subkomponente SINAMICS Integrated und ggf. über zusätzlich angeschlossene NX-Baugruppen.

Hinweis

Die SINUMERIK-NCU und ein SINAMICS Integrated befinden sich auf derselben Ebene im TIA Portal Openness-Objektmodell und erscheinen unter "DeviceComposition" als zwei unterschiedliche Geräte.

Um ein Telegramm zu konfigurieren, verwenden Sie "DriveObjectCollection". "DriveObjectCollection" ist ein Service des Antriebsobjekts (Geräteelement) vom aktuellen Kopfmodul (Geräteelement).

Um den Service "DriveObjectCollection" zu starten, navigieren Sie zu SINAMICS Integrated oder zum Kopfmodul der NX-Baugruppe. Die Hierarchie zwischen Geräten und Geräteelementen ist für SINAMICS Integrated und NX-Baugruppen identisch.

Das folgende Beispiel zeigt, wie Sie "DriveObjectCollection" über die Eigenschaft "Kopfmodul" finden:

DriveObjectCollection über Kopfmodul finden

```
foreach (Device device in project.Devices)
{
    foreach (DeviceItem deviceItem in device.DeviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.HM)
        {
            var driveObjectCollection = deviceItem.GetService<DriveObjectCollection>();
            // do something
        }
    }
}
```

Die SINUMERIK-NCU enthält ein SINAMICS Integrated Proxy-Objekt mit Referenzen auf SINAMICS Integrated.

Um auf ein SINAMICS Integrated-Gerät oder eine NX-Baugruppe zuzugreifen, navigieren Sie von der SINUMERIK-NCU über NCK zur DP Integrated-Schnittstelle. Anschließend ermitteln Sie das PROFIBUS-Mastersystem und navigieren zum verbundenen Slave.

Telegramme einfügen

Das folgende Beispiel zeigt, wie Sie ein Telegramm einfügen. Für den Zugriff benötigen Sie ein Antriebsobjekt.

Sie unterscheiden zwischen den Telegrammtypen mit Hilfe von ihren IDs.

Telegramm einfügen

```
using Siemens.Engineering.MC.DriveConfiguration;
TelegramComposition telegrams = drvObj.Telegrams;

//Create telegram
drvObj.Telegrams.Create(TelegramId.Supplementary_Telegram_701);

}
```

Mit Safety-Telegrammen arbeiten

Das folgende Beispiel zeigt, wie Sie ein Safety-Telegramm einfügen. Für den Zugriff benötigen Sie ein Antriebsobjekt.

Safety-Telegramm einfügen und auf Telegramm-Attribute zugreifen

```
using Siemens.Engineering.MC.DriveConfiguration;
```

Safety-Telegramm einfügen und auf Telegramm-Attribute zugreifen

```
TelegramComposition telegrams = drvObj.Telegrams;

//Add safety telegram
drvObj.Telegrams.Create(TelegramId.PROFIsafe_Standard_Telegram_30);

// Get and set safety telegram attributes
uint Failsafe_FDestinationAddress =
(uint)safetyTelegram.GetAttribute("Failsafe_FDestinationAddress");
uint Failsafe_FSourceAddress =
(uint)safetyTelegram.GetAttribute("Failsafe_FSourceAddress");
uint Failsafe_FIODBNumber = (uint)safetyTelegram.GetAttribute("Failsafe_FIODBNumber");
string Failsafe_FIODBName = safetyTelegram.GetAttribute("Failsafe_FIODBName").ToString();
uint Failsafe_FMonitoringtime =
(uint)safetyTelegram.GetAttribute("Failsafe_FMonitoringtime");
uint Failsafe_ManualAssignmentFIODBNumber =
(uint)safetyTelegram.GetAttribute("Failsafe_ManualAssignmentFIODBNumber");
bool Failsafe_ManualAssignmentFMonitoringtime =
(bool)safetyTelegram.GetAttribute("Failsafe_ManualAssignmentFMonitoringtime");

// Set safety telegram attributes
safetyTelegram.SetAttribute("Failsafe_ManualAssignmentFIODBNumber", 1);
safetyTelegram.SetAttribute("Failsafe_ManualAssignmentFMonitoringtime", true);
safetyTelegram.SetAttribute("Failsafe_FIODBNumber", 40000);
safetyTelegram.SetAttribute("Failsafe_FMonitoringtime", 200);
safetyTelegram.SetAttribute("Failsafe_FDestinationAddress", 15);

const int newSafetyTelegramId= 900;
if (safetyTgrm.CanChangeTelegram(newSafetyTelegramId)) {
safetyTgrm.TelegramId = newSafetyTelegramId; }
```

Das folgende Beispiel zeigt, wie Sie ein Safety-Telegramm löschen.

Safety-Telegramm löschen

```
using Siemens.Engineering.MC.DriveConfiguration;
//Remove Safety telegram
drvObj.Telegrams.DeleteTelegram(TelegramType.SafetyTelegram);
```

Telegramme erweitern

Das folgende Beispiel zeigt, wie Sie eine Erweiterung einfügen und die Größe eines Standardtelegramms verändern. Für den Zugriff benötigen Sie ein Antriebsobjekt.

Erweiterung einfügen und die Größe eines Standardtelegramms ändern

```
using Siemens.Engineering.MC.DriveConfiguration;
```

Erweiterung einfügen und die Größe eines Standardtelegramms ändern

```

TelegramComposition telegrams = drvObj.Telegrams;
Telegram telegram = telegrams.Find(TelegramType.MainTelegram);

Console.WriteLine("The Cu has the telegram: " + telegram.TelegramId);
Console.WriteLine("The Setpoint channel-specific size of the telegram is: "
+ telegram.GetOutputSize());

foreach (var address in telegram.Addresses)
{
    if (address.IoType == AddressIoType.Output)
    {
        Console.WriteLine("The Setpoint channel-specific IO start address of
the telegram on the connected PLC is: " + address.StartAddress);
    }
    else if (address.IoType == AddressIoType.Input)
    {
        Console.WriteLine("The Actual value channel-specific IO start address
of the telegram on the connected PLC is: " + address.StartAddress);
    }
}

// Create an additional telegram
if (drvObj.Telegrams.CreateAdditionalTelegram(2,4))
{
    drvObj.Telegrams.CreateAdditionalTelegram(2,4);
}

// Add a 3 word extension to the main telegram
Telegram mainTelegram == drvObj.Telegrams.Find(TelegramType.MainTelegram);
Int32 newSize = mainTelegram.GetSize(AddressIoType.Input) + 3;
if (mainTelegram.CanChangeSize(AddressIoType.Input, newSize, true))
{
    mainTelegram.ChangeSize(AddressIoType.Input, newSize, true)
}

```

5.21.6.12 Beispiele für Namespace Siemens.Engineering.MC.Drives**Telegramme vorbereiten**

Die Antriebskommunikation einer SINUMERIK-NCU erfolgt mithilfe von Telegrammen über die Subkomponente SINAMICS Integrated und ggf. über zusätzlich angeschlossene NX-Baugruppen.

Hinweis

Die SINUMERIK-NCU und ein SINAMICS Integrated befinden sich auf derselben Ebene im TIA Portal Openness-Objektmodell und erscheinen unter "DeviceComposition" als zwei unterschiedliche Geräte.

Um ein Telegramm zu konfigurieren, verwenden Sie "DriveObjectContainer". "DriveObjectContainer" ist ein Service des Antriebsobjekts (Geräteelement) vom aktuellen Kopfmodul (Geräteelement).

Um den Service "DriveObjectContainer" zu starten, navigieren Sie zu SINAMICS Integrated oder zum Kopfmodul der NX-Baugruppe. Die Hierarchie zwischen Geräten und Geräteelementen ist für SINAMICS Integrated und NX-Baugruppen identisch.

Das folgende Beispiel zeigt, wie Sie "DriveObjectContainer" über die Eigenschaft "Kopfmodul" finden:

DriveObjectContainer über Kopfmodul finden

```
foreach (Device device in project.Devices)
{
    foreach (DeviceItem deviceItem in device.DeviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.HM)
        {
            var driveObjectContainer = deviceItem.GetService<DriveObjectContainer>();
            // do something
        }
    }
}
```

Die SINUMERIK-NCU enthält ein SINAMICS Integrated Proxy-Objekt mit Referenzen auf SINAMICS Integrated.

Um auf ein SINAMICS Integrated-Gerät oder eine NX-Baugruppe zuzugreifen, navigieren Sie von der SINUMERIK-NCU über NCK zur DP Integrated-Schnittstelle. Anschließend ermitteln Sie das PROFIBUS-Mastersystem und navigieren zum verbundenen Slave.

Telegramme einfügen und löschen

Das folgende Beispiel zeigt, wie Sie ein Telegramm einfügen. Für den Zugriff benötigen Sie ein Antriebsobjekt.

Telegramm einfügen und auf Telegramm-Attribute zugreifen

```
using Siemens.Engineering.MC.Drives;
```

Telegramm einfügen und auf Telegramm-Attribute zugreifen

```
TelegramComposition telegrams = drvObj.Telegrams;

//Add telegram
drvObj.Telegrams.InsertAdditionalTelegram(tgrmNumber);

//Find telegram
Telegram telegram = drvObj.Telegrams.Find(TelegramType.MainTelegram);

//Add safety telegram
const int tgrmNumber = 30;
drvObj.Telegrams.InsertSafetyTelegram(tgrmNumber) (

//Find safety telegram
Telegram safetyTgrm = drvObj.Telegrams.Find(TelegramType.SafetyTelegram);

// Get and set safety telegram attributes
uint watchDogTime =
(uint)safetyTgrm.GetAttribute("Failsafe_FMonitoringtime");

safetyTgrm.SetAttribute("Failsafe_FMonitoringtime", 300);

const int newSafetyTelegramNumber= 902;
if (safetyTgrm.CanChangeTelegram(newSafetyTelegramNumber)) {
safetyTgrm.TelegramNumber = newSafetyTelegramNumber; }

//Add supplementary telegram
const int tgrmNumber = 701;
drvObj.Telegrams.InsertSupplementaryTelegram(tgrmNumber);

Telegram telegram =
drvObj.Telegrams.Find(TelegramType.SupplementaryTelegram);
```

Das folgende Beispiel zeigt, wie Sie ein Telegramm löschen.

Telegramm löschen

```
using Siemens.Engineering.MC.Drives;

//Remove safety telegram
drvObj.Telegrams.EraseTelegram(TelegramType.SafetyTelegram);

//Remove supplementary telegram
drvObj.Telegrams.EraseTelegram(TelegramType.SupplementaryTelegram);
```

Hinweis

Sie können ein Haupttelegramm (MainTelegram) ändern, aber nicht löschen.

Mit Safety-Telegrammen arbeiten

Das folgende Beispiel zeigt, wie Sie ein Safety-Telegramm einfügen. Für den Zugriff benötigen Sie ein Antriebsobjekt.

Safety-Telegramm einfügen und auf Telegramm-Attribute zugreifen

```
using Siemens.Engineering.MC.Drives;
TelegramComposition telegrams = drvObj.Telegrams;

//Add safety telegram
const int tgrmNumber = 30;
drvObj.Telegrams.InsertSafetyTelegram(tgrmNumber);

//Find safety telegram
Telegram safetyTgrm = drvObj.Telegrams.Find(TelegramType.SafetyTelegram);

// Get and set safety telegram attributes
uint Failsafe_FDestinationAddress =
(uint)safetyTelegram.GetAttribute("Failsafe_FDestinationAddress");
uint Failsafe_FSourceAddress =
(uint)safetyTelegram.GetAttribute("Failsafe_FSourceAddress");
uint Failsafe_FIODBNumber = (uint)safetyTelegram.GetAttribute("Failsafe_FIODBNumber");
string Failsafe_FIODBName = safetyTelegram.GetAttribute("Failsafe_FIODBName").ToString();
uint Failsafe_FMonitoringtime =
(uint)safetyTelegram.GetAttribute("Failsafe_FMonitoringtime");
uint Failsafe_ManualAssignmentFIODBNumber =
(uint)safetyTelegram.GetAttribute("Failsafe_ManualAssignmentFIODBNumber");
bool Failsafe_ManualAssignmentFMonitoringtime =
(bool)safetyTelegram.GetAttribute("Failsafe_ManualAssignmentFMonitoringtime");

// Set safety telegram attributes
safetyTelegram.SetAttribute("Failsafe_ManualAssignmentFIODBNumber", 1);
safetyTelegram.SetAttribute("Failsafe_ManualAssignmentFMonitoringtime", true);
safetyTelegram.SetAttribute("Failsafe_FIODBNumber", 40000);
safetyTelegram.SetAttribute("Failsafe_FMonitoringtime", 200);
safetyTelegram.SetAttribute("Failsafe_FDestinationAddress", 15);

const int newSafetyTelegramNumber= 900;
if (safetyTgrm.CanChangeTelegram(newSafetyTelegramNumber)) {
safetyTgrm.TelegramNumber = newSafetyTelegramNumber; }
```

Das folgende Beispiel zeigt, wie Sie ein Safety-Telegramm löschen.

Safety-Telegramm löschen

```
using Siemens.Engineering.MC.Drives;
//Remove Safety telegram
drvObj.Telegrams.EraseTelegram(TelegramType.SafetyTelegram);
```

Telegramme erweitern

Das folgende Beispiel zeigt, wie Sie eine Erweiterung einfügen und die Größe eines Standardtelegramms verändern. Für den Zugriff benötigen Sie ein Antriebsobjekt.

Erweiterung einfügen und die Größe eines Standardtelegramms ändern

```
using Siemens.Engineering.MC.Drives;
```

Erweiterung einfügen und die Größe eines Standardtelegramms ändern

```

TelegramComposition telegrams = drvObj.Telegrams;
Telegram telegram = telegrams.Find(TelegramType.MainTelegram);

Console.WriteLine("The Cu has the telegram: " + telegram.TelegramNumber);
Console.WriteLine("The Setpoint channel-specific size of the telegram is: "
+ telegram.GetOutputSize());

foreach (var address in telegram.Addresses)
{
    if (address.IoType == AddressIoType.Output)
    {
        Console.WriteLine("The Setpoint channel-specific IO start address of
the telegram on the connected PLC is: " + address.StartAddress);
    }
    else if (address.IoType == AddressIoType.Input)
    {
        Console.WriteLine("The Actual value channel-specific IO start address
of the telegram on the connected PLC is: " + address.StartAddress);
    }
}

// Add an additional telegram
if (drvObj.Telegrams.CanInsertAdditionalTelegram(2,4))
{
    drvObj.Telegrams.InsertAdditionalTelegram(2,4);
}

// Add a 3 word extension to the main telegram
Telegram mainTelegram == drvObj.Telegrams.Find(TelegramType.MainTelegram);
Int32 newSize = mainTelegram.GetSize(AddressIoType.Input) + 3;
if (mainTelegram.CanChangeSize(AddressIoType.Input, newSize, true))
{
    mainTelegram.ChangeSize(AddressIoType.Input, newSize, true)
}

```

5.22 Funktionen für SINUMERIK MC

5.22.1 Einführung

Für SINUMERIK MC ist in Openness nur die Navigation auf den PLC Software-Container verfügbar.

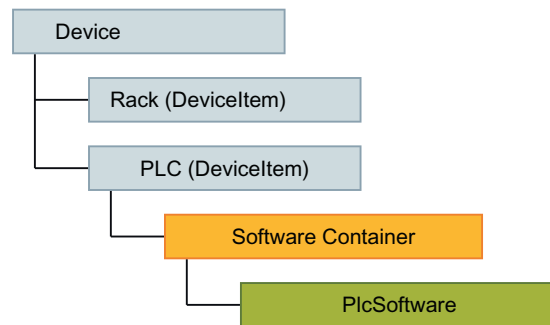
Weitere Openness Funktionalitäten sind weder für die steuerungsspezifische Anteile noch für die PC-spezifische Anteile einer SINUMERIK MC verfügbar.

5.22.2 Codebeispiel

5.22.2.1 Auf PLC-Software zugreifen

Übersicht

Das folgende Diagramm beschreibt die Darstellung des Software-Containers und der PLC-Software im Objektmodell:



Das folgende Code-Beispiel zeigt, wie Sie auf eine PLC, unabhängig von konkreter Realisierung (integrierte SINUMERIK PLC, SIMATIC PLC, Software PLC im PC) anhand der Eigenschaft "CPU" zugreifen:

PLCs finden

```

Device ncuDevice = ...
DeviceItem plc = GetPlc(mcuDevice.DeviceItems);

...
DeviceItem GetPlc(DeviceItemComposition deviceItems)
{
    if (deviceItems.Count == 0)
    {
        return null;
    }
    foreach (var deviceItem in deviceItems)
    {
        if (deviceItem.Classification == DeviceItemClassifications.CPU)
            return deviceItem;

        DeviceItem plc = GetPlc(deviceItem.DeviceItems);
        if (plc != null)
            return plc;
    }

    return null;
}

```

Für den konkreten Zugriff auf den PLC-Software-Container lesen Sie weiter unter "Auf Software-Ziel zugreifen".

5.23 Funktionen für Startdrive

5.23.1 Einführung

Über TIA Portal-Openness automatisieren Sie das Engineering und steuern das TIA Portal über ein von Ihnen selbst erstelltes Programm.

Für dieses selbst erstellte Programm finden Sie in dieser Hilfe viele Informationen und Beispielcodes. Auch für die TIA Portal-Anwendung "Startdrive" können Sie eigene Programme zusammenstellen und anwenden.

Bevor Sie für Startdrive aus den nachfolgend gelisteten Beispielcodes ein eigenes Programm zusammenstellen, beachten Sie bitte die übergreifenden Informationen zu Openness, die Sie unter folgenden Stichworten in diesem Informationssystem finden:

- Voraussetzungen für TIA Portal Openness
- TIA Portal Openness installieren
- Auf das TIA Portal zugreifen
- TIA Portal Openness-Objektmodell
- Programmierschritte

Hinweis

Aktivierter Projektschutz (UMAC)

Falls Sie ein Projekt über "Openness API" bearbeiten wollen, für das ein Projektschutz aktiviert ist, benötigen Sie dazu die entsprechenden Zugriffsrechte. Ihr Benutzerkonto muss folgende Funktionsrechte besitzen:

- Projekt über Openness API ändern
- Projekt öffnen und bearbeiten
- Projekt lesend öffnen

Für Technologieobjekte werden weitere Funktionsrechte notwendig. Falls Sie keine Zugriffsrechte besitzen, kontaktieren Sie Ihren Administrator.

Für Antriebe ab SINAMICS FW V6.1 oder höher benötigen Sie für Ihr Benutzerkonto beim Online-Zugriff auf den Antrieb die gleichen Zugriffsrechte wie in Startdrive.

Weitergehende Informationen zu Zugriffsrechten und zur Benutzerverwaltung allgemein finden Sie auf der Seite "Benutzerverwaltung und Security".

5.23.2 Security-Hinweise

5.23.2.1 Verschlüsselte Kommunikation mit SecureString-Passwörtern

Hinweis

SecureString-Passwörter zur sicheren Kommunikation

Um bei der Verwendung von Startdrive Openness-Funktionen die Kommunikation zwischen der Openness API und dem TIA Portal abzusichern, verwenden Sie mit einem SecureString verschlüsselte Passwörter.

Hinweis

Kommunikationsrichtung

Die Verwendung von SecureString-Passwörtern ist nur beim Download von Openness-Funktionen zum TIA Portal verfügbar und nicht in umgekehrter Kommunikationsrichtung.

Hinweis

Ein SecureString-Objekt sollte nie aus einem String erstellt werden, da die sensiblen Daten bereits den Auswirkungen der Speicherpersistenz der unveränderlichen Stringklasse unterliegen. Die beste Methode zum Erstellen eines SecureString-Objekts ist eine nicht verwaltete Zeichenfolge, z. B. die Console.ReadKey-Methode.

Hinweis

Der API-Benutzer ist verantwortlich für die Sicherheitsmaßnahmen zur Handhabung von Passwörtern durch Code.

Weitere Informationen zur Konfiguration von SecureString-Passwörtern finden Sie in den Codebeispielen Download (Seite 1157) und SecureString-Passwort vergeben (Seite 1185).

Siehe auch

Passwörter in TIA Openness (Seite 1129)

5.23.2.2 Passwörter in TIA Openness

Hantierung von Passwörtern in TIA Openness

Das Passwort-Handling von Startdrive Openness basiert auf TIA Openness.

Beachten Sie bitte die übergreifenden Informationen zu Passwörtern in TIA Openness, die Sie unter folgenden Stichworten in dieser Hilfe finden:

- Display-Passwort festlegen
- Handhabung bausteingebundener PLC-Passwörter

- PLC durch Passwort schützen
- Einstellung der Zugriffsstufen
- Einen Baustein schützen und Schutz aufheben
- Hardware, Software und Dateien in PLC-Gerät laden

Siehe auch

Verschlüsselte Kommunikation mit SecureString-Passwörtern (Seite 1129)

5.23.3 Typidentifizier - Kennzeichnung der Komponenten

Jede Ausprägung einer beliebigen Startdrive-Komponente besitzt eine eindeutige Nummer, die als Typidentifizier bezeichnet wird. Im Openness-Programmcode können Sie diesen Typidentifizier nutzen, um eine Komponente eindeutig zu identifizieren und zu benennen.

In Startdrive ist die Anzeige des Typidentifiziers optional und standardmäßig deaktiviert.

Anzeige des Typidentifiziers in Startdrive aktivieren

1. Wählen Sie in der Startdrive-Projektansicht das Menü "Extras > Einstellungen". Der Konfigurationsbereich "Einstellungen" wird geöffnet.
2. Wählen Sie in der Sekundärnavigation den Eintrag "Hardware-Konfiguration".
3. Aktivieren Sie die Option "Anzeige des Type Identifiers für Geräte und Module aktivieren". Die Anzeige des Typidentifiziers ist nun aktiv.

Typidentifizier in Startdrive auslesen

Der Typidentifizier kann in einem Startdrive-Projekt bei aktivierter Anzeige an folgenden Stellen ausgelesen werden:

- Für alle Komponenten (im Inspektorfenster):
- Für Control Units (beim Anlegen eines Antriebsgeräts)

Um den Typidentifizier für eine Komponente im Inspektorfenster auszulesen, gehen Sie so vor:

1. Doppelklicken Sie in der Gerätesicht des Startdrive-Projekts auf die gewünschte Komponente.
Das Inspektorfenster wird geöffnet. Die aktive Ausprägung der Komponente wird in der Liste angezeigt.
2. Den dazu gehörenden Typidentifizier finden in der äußeren rechten Spalte.
Beispiel: `OrderNumber: 6SL3131-7TE23-6Axx`
Kopieren Sie diesen Typidentifizier in Ihre Openness-Anwendung.

Um den Typidentifizier für eine Control Unit auszulesen, gehen Sie so vor:

1. Doppelklicken Sie in der Startdrive-Projektnavigation auf "Neues Gerät hinzufügen". Der gleichnamige Dialog wird geöffnet.
2. Selektieren Sie die gewünschte Regelungsbaugruppe aus der Auswahl. Rechts in der Detailanzeige wird nun der Typidentifizier (unter der Artikelnummer und Firmware-Nummer) für die entsprechende Control Unit angezeigt.
Beispiel: OrderNumber: 6SL3040-1MA01-0Axx/V5.2/S120
3. Kopieren Sie diesen Typidentifizier in Ihre Openness-Anwendung.

5.23.4 Referenzen

5.23.4.1 AddressComposition

AddressComposition

Die Klasse `AddressComposition` repräsentiert die Adresse eines Telegramms.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public sealed class AddressComposition
```

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
<code>IoType</code>	<code>AddressIoType</code> (Seite 1132)	Gibt Informationen zum Typ der Adresse zurück.
<code>Context</code>	<code>AddressContext</code> (Seite 1132)	Gibt Informationen zum Kontext der Adresse zurück.
<code>StartAddress</code>	<code>Int32</code>	Gibt die Startadresse des Telegramms zurück oder legt diese fest.
<code>Length</code>	<code>Int32</code>	Gibt die Länge des Telegramms zurück.

Siehe auch

`TelegramType` (Seite 1150)

5.23.4.2 AddressContext

AddressContext

Das Enum AddressContext enthält Informationen zum Kontext der Adresse.

Namespace: Siemens.Engineering.MC.Drives

Assembly: Siemens.Engineering.MC.Drives in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public enum AddressContext
```

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
AddressContext.None	Für die Adresse wurde kein Kontext gefunden
AddressContext.Device	Kontext ist eine Geräteadresse
AddressContext.Head	Kontext ist eine Kopfadresse

5.23.4.3 AddressIoType

AddressIoType

Das Enum AddressIoType enthält Informationen zum Typ der Adresse.

Namespace: Siemens.Engineering.MC.Drives

Assembly: Siemens.Engineering.MC.Drives in Siemens.Engineering.dll

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public enum AddressIoType
```

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
AddressIoType.None	Der IO-Typ ist nicht anwendbar
AddressIoType.Input	Typ ist eine Eingangsadresse
AddressIoType.Output	Typ ist eine Ausgangsadresse
AddressIoType.Diagnosis	Typ ist eine Diagnoseadresse
AddressIoType.Substitute	Typ ist eine Ersatzadresse

5.23.4.4 Commissioning

Commissioning

Die Klasse `Commissioning` dient der Inbetriebnahme eines Antriebs. Sie kann auf `DriveFunctionInterface` oder `OnlineDriveFunctionInterface` angewendet werden. Wenn der Anwender die Funktion für einen SINAMICS S-Antrieb verwendet, gibt sie den Rückgabewert null zurück.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Parameter	Rückgabewert	Beschreibung
<code>SetSimoGearMlfb</code>	<code>string simoGearMlfb</code>	<code>bool</code>	Stellt die SIMOGEAR-Artikelnummer eines Antriebs ein.

Siehe auch

SIMOGEAR-Artikelnummer für G115D-Antriebe setzen (Seite 1188)

5.23.4.5 ConfigurationEntry

ConfigurationEntry

Die Klasse `ConfigurationEntry` dient zum Speichern von Parameterdaten, die aus den `ConfigurationEntryCompositions` einer Motor- oder Geberkonfiguration ermittelt werden können.

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
<code>EnumValueList</code>	<code>IDictionary<it, string></code>	Gibt eine Liste mit möglichen Werten des enum-Parameters zurück.
<code>MaxValue</code>	<code>object</code>	Maximalwert des <code>ConfigurationEntry</code> .
<code>MinValue</code>	<code>object</code>	Minimalwert des <code>ConfigurationEntry</code> .
<code>Name</code>	<code>string</code>	Name des <code>ConfigurationEntry</code> .
<code>Number</code>	<code>int</code>	Numerische Darstellung des Namens für <code>ConfigurationEntry</code> .
<code>Description</code>	<code>string</code>	Beschreibung des <code>ConfigurationEntry</code> .
<code>Parent</code>	<code>IEngineeringObject</code>	Engineering Object Model-Elternelement dieses Objekts.
<code>Unit</code>	<code>string</code>	Einheit des <code>ConfigurationEntry</code> .
<code>Value</code>	<code>object</code>	Wert des <code>ConfigurationEntry</code> .

5.23.4.6 DriveDataEncryption (ab SINAMICS FW V6.1)

Die Klasse `DriveDataEncryption` ermöglicht, die Verschlüsselung der Antriebsdaten (DDE, Drive Data Encryption) für SINAMICS-Antriebe ab Firmware-Version V6.1 mit Openness zu aktivieren und zu deaktivieren.

Die folgende Tabelle beschreibt die **Methoden** der Klasse `DriveDataEncryption`:

Name	Parameter	Rückgabewert	Beschreibung	Ausnahmen
Activate	SecureString password	bool	<code>DriveDataEncryption</code> mit Passwort aktivieren.	-
Deactivate	SecureString password	bool	<code>DriveDataEncryption</code> mit Passwort deaktivieren.	-

Hinweis

Klasse `Security`

Die Klasse `Security` ist eine übergeordnete Klasse. Die Klasse wird im Hintergrund verwendet, um auf alle APIs zuzugreifen, die mit `Security` zu tun haben.

5.23.4.7 DriveDomainFunctions

DriveDomainFunctions

Die Klasse `DriveDomainFunctions` dient der Wiederherstellung der Werkseinstellungen bzw. der Sicherung des RAM-Inhalts auf das ROM.

Sie kann nur auf ein `OnlineDriveFunctionInterface`-Objekt angewandt werden.

Bei G120-Antrieben kann nur auf das `DriveDomainFunctions`-Objekt zugegriffen werden, wenn das Power Module mit dem Gerät verkabelt ist. Andernfalls wird null oder eine Exception zurückgegeben.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
<code>PerformFactoryReset</code>	Diese Methode ist für die Wiederherstellung der Werkseinstellungen zuständig.
<code>PerformRAMtoROMCopyAllDriveObject</code>	Die Daten aller Antriebsobjekte werden aus dem RAM auf die Speicherkarte/Festplatte geschrieben.

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
<code>Parent</code>	<code>IEngineeringObject</code>	Engineering Object Model-Elternelement dieses Objekts.

5.23.4.8 DriveObject

DriveObject

Die Klasse `DriveObject` ermöglicht den Zugriff auf das Antriebsobjekt. Über das Antriebsobjekt ist z. B. der Zugriff auf Antriebsparameter oder das Telegramm möglich.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public sealed class DriveObject
```

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
Parameters	DriveParameter-Composition (Seite 1139)	Gibt eine Liste mit den verfügbaren Parametern des Antriebsobjekts zurück.
Telegrams	TelegramComposition (Seite 1149)	Gibt eine Liste mit den verfügbaren Telegrammen des Antriebsobjekts zurück. Die Liste kann mit der Klasse <code>TelegramComposition</code> verändert werden.

Siehe auch

Antriebsobjekt ermitteln (Seite 1155)

5.23.4.9 DriveObjectActivation

DriveObjectActivation

Die Klasse `DriveObjectActivation` dient der Aktivierung der Module bzw. der Ermittlung des Modulstatus. Sie kann auf `DriveObjectFunctions` von `DriveFunctionInterface` oder `OnlineDriveFunctionInterface` angewendet werden.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
<code>ChangeActivationState(DriveObjectActivationState)</code>	Ändert den Aktivierungsstatus des Antriebsobjekts. Gibt false zurück, wenn die Operation nicht abgeschlossen werden kann. Mögliche Statuswerte: • Deactivate • Activate • DeactivateAndNotPresent

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
ActivationState	DriveObjectActivationState	Gibt den aktuellen Statuswert zurück.
IsActive	Boolean	Gibt bei aktivem Antriebsobjekt true zurück.

5.23.4.10 DriveObjectContainer

DriveObjectContainer

Der `DriveObjectContainer` ist ein Service des Antriebsobjekts (`DeviceItem`) vom aktuellen Gerät (`Device`).

Die folgende Tabelle beschreibt die **Navigatoren** des `DriveObjectContainer`:

Name	Datentyp	Beschreibung
DriveObjects	DriveObjectComposition (Seite 1139)	Gibt eine Liste mit den verfügbaren Antriebsobjekten zurück. Die Antriebsobjekte ermöglichen den Zugriff auf Antriebsparameter und Telegramme.
Parent	IEngineeringObject	Engineering Object Model-Elternelement dieses Objekts.

5.23.4.11 ModuleAccessPoint

ModuleAccessPoint

Der `ModuleAccessPoint` ist ein Service des Antriebsobjekts (`DeviceItem`) vom aktuellen Gerät (`Device`).

Die folgende Tabelle beschreibt die **Eigenschaften** des Hardware Identifier des `ModuleAccessPoint`:

Name	Datentyp	Zugriff	Beschreibung
Hardware Identifier	IEnumerable<IBrowsable>	ReadOnly	Gibt Liste der Hardware Identifier zurück.

5.23.4.12 DriveObjectTypeHandler

DriveObjectTypeHandler

Die Klasse `DriveObjectTypeHandler` dient der Umstellung des Antriebsobjekt-Typs für jedes Antriebsobjekts und der Ermittlung des Antriebsobjekt-Typs sowie aller möglichen Antriebsobjekt-Typen des aktuellen Antriebsobjekts. Sie kann nur auf `DriveFunctionInterface` angewendet werden.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
ChangeDriveObjectType ((target) DriveObjectType)	Ändert den aktuellen Typ des Antriebsobjekts auf einen auswählbaren neuen Typ. Gibt false zurück, wenn die Operation nicht abgeschlossen werden kann.

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
PossibleDriveObjectTypes	DriveObjectTypeComposition	Die Verwendung von targetDriveObjectType führt auf dem aktuellen Antriebsobjekt zu einer Exception.
CurrentDriveObjectType	DriveObjectType	Zeigt den aktuell zugewiesenen Antriebsobjekt-Typ.

5.23.4.13 DriveParameter

DriveParameter

Die Klasse `DriveParameter` ermöglicht den Zugriff auf einen Antriebsparameter. Nicht alle Antriebsparameter sind für den Zugriff über Openness verfügbar.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public sealed class DriveParameter
```

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
ArrayIndex	Int32	Gibt den Index eines Array-Parameters zurück. Wertebereich: 0-7FFF Bei Parametern ohne Array ist der Index -1 Beispiel <code>p108[4].15</code> <code>par.ArrayIndex</code> ergibt 4
ArrayLength	Int32	Gibt die Anzahl der Array Elemente zurück. Bei Parametern ohne Array ist der Wert 0

Name	Datentyp	Beschreibung
Bits	DriveParameter-Composition (Seite 1139)	Gibt ein Objekt <code>DriveParameter</code> für ein Bit des Parameters zurück. Damit kann von einem Bitparameter z. B. der Wert oder der Name des Bitparameters gelesen werden. Beispiel <pre>DriveParameter param133 = cu.Parameters.Find(133, 0); DriveParameter param133Bit1 = param133.Bits[1]; String paramName = param133Bit1.Name;</pre>
EnumValueList	IDictionary<int, string>	Gibt eine Liste mit möglichen Werten des enum Parameters zurück. Z. B. <1, [1] Quick commissioning> null, wenn der Parameter nicht vom Typ <code>Enum</code> ist.
MaxValue	Object	Gibt den maximalen Wert für die aktuell gewählte Einheit zurück.
MinValue	Object	Gibt den minimalen Wert für die aktuell gewählte Einheit zurück.
Name	string	Gibt den Namen des Parameters zurück. Z. B. "p108[0].2"
ParameterText	string	Gibt den Text der Kurzbeschreibung für den Parameter zurück.
Number	Int32	Gibt die Nummer des Parameters zurück. Beispiel p108[0].2 Rückgabewert ist 108
Unit	string	Gibt die Einheit des Parameters als Text zurück.
Value	Object	Gibt den Offline/Online Wert des Parameters zurück oder schreibt einen Wert auf den Parameter. Bei Schreibfehlern wird eine <code>EngineeringTargetInvocationException</code> ausgelöst. Beispiele - P2080Bit6.Value = 0; - P2080Bit6.Value = cu.Parameters.Find("r19"); BICO-Quelle Der Parameter einer BICO-Quelle kann nur gelesen werden BICO-Signalsenken Mögliche Werte sind 0, 1 oder ein Objekt <code>DriveParameter</code> . Ein Objekt <code>DriveParameter</code> wird zurückgegeben, wenn die BICO-Signalsenke mit einem anderen Parameter verbunden ist. Siehe auch Beispiel BICO-Parameter lesen und schreiben (Seite 1156).

Siehe auch

Parameter lesen und schreiben (Seite 1182)

5.23.4.14 DriveParameterComposition**DriveParameterComposition**

Die Klasse `DriveParameterComposition` ermöglicht den Zugriff auf Parameter des Antriebs. Nicht alle Antriebsparameter sind für den Zugriff über Openness zugelassen.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public sealed class DriveParameterComposition
```

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
<code>Find(string)</code>	Gibt das Objekt <code>DriveParameter</code> (Seite 1137) zurück, das über den Namen gesucht wird. null, wenn der Parameter nicht gefunden wird Beispiel <code>Find("P108[1]");</code>
<code>Find(UInt16, Int32)</code>	Gibt das Objekt <code>DriveParameter</code> (Seite 1137) zurück, das über den Parameter- und Array-Index gesucht wird. null, wenn der Parameter nicht gefunden wird Beispiele <code>cu.Find(108, 1);</code> <code>cu.Find(51, -1);</code>

5.23.4.15 HardwareProjection

Die Klasse `HardwareProjection` ist verantwortlich für die Inbetriebnahme des Motors und des Gebers. Das Objekt kann bei `DriveFunctionInterface` und `OnlineDriveFunctionInterface` gefunden werden.

Die Projektierung kann bei G120-Antrieben für Motoren und Geber online wie offline durchgeführt werden. Bei S120-Antrieben ist die Projektierung dagegen nur offline möglich.

Bei G120-Antrieben kann auf das Objekt `HardwareProjection` nur zugegriffen werden, wenn das Power Module im Antriebsgerät eingesteckt ist. Andernfalls wird beim Aufruf der Funktionen `HardwareProjection` eine null oder eine Exception zurückgegeben.

5.23 Funktionen für Startdrive

Bei einer Offline-Projektierung nutzen Sie die Hardware-Projektierung des `DriveFunctionInterface`. Bei einer Online-Projektierung nutzen Sie Hardware-Projektierung des `OnlineDriveFunctionInterface`.

Namespace: `Siemens.Engineering.MC.Drives`

`Siemens.Engineering.MC.Drives.DFI`

`Siemens.Engineering.MC.Drives.Enums`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Parameter	Rückgabewert	Beschreibung	Ausnahmen
<code>SetMotorType</code>	<code>MotorType type</code> , <code>ushort driveDataSet</code>	bool	Stellt den Motor-Typ an der Control Unit ein (nur bei G120).	Nür für G120-Antriebe verfügbar.
<code>GetCurrentMotorConfiguration</code> Für S120: <code>GetCurrentMotorConfiguration(ushort motorDataset)</code>	<code>ushort driveDataSet</code>	<code>MotorConfiguration</code>	Ermittelt abhängig von der Datensatznummer des Antriebs den aktuell existierenden Konfigurationsbereich.	-
<code>ProjectMotorConfiguration</code>	<code>MotorConfiguration motConfig</code> , <code>ushort driveDataSet</code>	bool	Projektiert die Motorkonfiguration eines Antriebsgeräts abhängig von der Datensatznummer des Antriebs.	-
<code>SetEncoder</code>	<code>EncoderType type</code> , <code>EncoderInterface interfaceType</code> , <code>AbsoluteIncrementalFlag absIncFlag</code> , <code>RotaryLinearFlag rotLinFlag</code> , <code>ushort encoderNumber</code>	bool	Stellt den Geber (Encoder) an der Control Unit ein (nur bei G120).	Nür für G120-Antriebe verfügbar.
<code>GetCurrentEncoderConfiguration</code>	<code>ushort encoderNumber</code>	<code>EncoderConfiguration</code>	Ermittelt abhängig von der Datensatznummer des Gebers den aktuell existierenden Konfigurationsbereich.	-
<code>ProjectEncoderConfiguration</code>	<code>EncoderConfiguration encConfig</code> , <code>ushort encoderNumber</code>	bool	Projektiert die Motorkonfiguration eines Antriebsgeräts abhängig von der Datensatznummer des Gebers (Encoders).	-

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriff	Beschreibung
Parent	<code>IEngineeringObject</code>	ReadOnly	Engineering Object Model-Eltern-element dieses Objekts.

Siehe auch

Technologieobjekte über Telegrammobjekte mit Hardware-Modulen verbinden (Seite 1191)

5.23.4.16 EncoderConfiguration

EncoderConfiguration

Die Klasse `EncoderConfiguration` speichert Daten von Nicht-Siemens-Gebern (-Encodern).

- Das Objekt `ConfigurationEntryComposition` muss vom Anwender ausgefüllt werden.
- Das Objekt `RequiredConfigurationEntries` muss ebenfalls ausgefüllt werden.

Namespace: `Siemens.Engineering.MC.Drives`
`Siemens.Engineering.MC.Drives.DFI`
`Siemens.Engineering.MC.Drives.Enum`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriff	Beschreibung
<code>RequiredConfigurationEntries</code>	<code>ConfigurationEntryComposition</code>	ReadOnly	Zugängliche Konfigurationseingänge von <code>EncoderConfiguration</code> .
<code>Parent</code>	<code>IEngineeringObject</code>	ReadOnly	Engineering Object Model-Elternelement dieses Objekts.
<code>RequiredConfigurationEntries</code>	<code>IEnumerable<IBrowsable></code>	Writeable	Gibt alle verwendbaren Geber-Parameter zurück, mit Ausnahme von p404.0 und/oder p404.1.
<code>EncoderTypes</code>	<code>IEnumerable<IBrowsable></code>	ReadOnly	Gibt je nach Geber die Parameter p404.0 oder/und p404.1 zurück.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Parameter	Rückgabewert	Beschreibung
<code>SetEncoderType</code>	<code>RotaryLinearFlag</code> enum	bool	Legt den Gebertyp des nicht-DRIVECLiQ-Gebers auf rotatorisch oder linear fest.
<code>SetEncoderType</code>	<code>RotaryLinearFlag</code> enum, <code>AbsoluteIncrementalFlag</code> enum	bool	Legt den Gebertyp des DRIVE-CLiQ-Gebers auf rotatorisch oder linear sowie absolut oder inkrementell fest. Nicht zur Verwendung mit nicht-DRIVE-CLiQ-Gebern vorgesehen. G220-Antriebe mit Firmware-Version V6.2 unterstützen nur den Gebertyp absolut oder inkrementell. Diese API kann zur Einstellung verwendet werden.

5.23.4.17 MotorConfiguration

MotorConfiguration

Die Klasse `MotorConfiguration` ist für die Inbetriebnahme von Motoren und Gebern verantwortlich.

Namespace: `Siemens.Engineering.MC.Drives`
`Siemens.Engineering.MC.Drives.DFI`
`Siemens.Engineering.MC.Drives.Enums`
Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Verhalten bei Siemens-Motoren:

Um den Motor-Typ einzustellen muss die Aktion `SetMotortype` aufgerufen werden. Diese Aktion besteht aus der Enum `MotorType` und einer Nummer, die die `DriveDatasetNumber` repräsentiert.

Die Aktion `SetMotortype` steht für S120-Antriebe nicht zur Verfügung. Bei diesen Antrieben wird der Motortyp über den Hardware-Katalog durch die Methode `PlugNew()` eingestellt.

Die folgende Tabelle beschreibt die Enums:

Enum-Name	Beschreibung
<code>NoMotor</code>	0: No motor
<code>InductionMotor</code>	1: Induction motor
<code>SynchronousMotor</code>	2: Synchronous motor
<code>NoCodeNumber1LE1InductionMotor</code>	10: 1LE1 induction motor (not a code number)
<code>NoCodeNumber1LG6InductionMotor</code>	13: 1LG6 induction motor (not a code number)
<code>NoCodeNumber1xx1SIMOTICSFDInductionMotor</code>	14: 1xx1 SIMOTICS FD induction motor (not a code number)
<code>NoCodeNumber1LA7InductionMotorNoCodeNumber</code>	17: 1LA7 induction motor (not a code number)
<code>MotorSeriesNumber1LA81PQ8StandardInduction</code>	18: 1LA8 / 1PQ8 standard induction motor series
<code>NoCodeNumber1LA9InductionMotor</code>	19: 1LA9 induction motor (not a code number)

Verhalten bei Nicht-Siemens-Motoren:

Die Klasse `MotorConfiguration` wird zum Speichern von Motordaten von Nicht-Siemens-Motoren benutzt. Es enthält 2 Objekte `ConfigurationEntryComposition`, die bei der Inbetriebnahme mit den entsprechenden Daten des Motors befüllt werden müssen. Das Objekt `RequiredConfigurationEntries` muss ebenfalls ausgefüllt werden. Das Objekt `OptionalConfigurationEntries` muss nicht ausgefüllt werden.

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
RequiredConfigurationEntries	ConfigurationEntryComposition	Zugängliche Konfigurationseingänge dieser Motorkonfiguration.
OptionalConfigurationEntries	ConfigurationEntryComposition	Zugängliche Konfigurationseingänge dieser Motorkonfiguration.
Parent	IEngineeringObject	Engineering Object Model-Elternelement dieses Objekts (nur S120-Antriebe).

5.23.4.18 OnlineDriveObject

OnlineDriveObject

Die Klasse `OnlineDriveObject` ermöglicht den Online-Zugriff auf das Antriebsobjekt. Über das Antriebsobjekt ist der Zugriff auf Antriebsparameter möglich.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public sealed class OnlineDriveObject
```

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
Parameters	DriveParameterComposition (Seite 1139)	Gibt eine Liste mit den verfügbaren Parametern des Online-Antriebsobjekts zurück. null, wenn der Modus "Offline" ist. Im Offline-Modus wird beim Aufruf einer Methode oder bei einem Schreibzugriff auf einen Parameter eine Exception ausgelöst.

Siehe auch

Antriebsobjekt ermitteln (Seite 1155)

Parameter lesen und schreiben (Seite 1182)

Parameter online lesen und schreiben (Seite 1184)

5.23.4.19 OnlineDriveObjectContainer

OnlineDriveObjectContainer

Der `OnlineDriveObjectContainer` ist ein Service des Antriebsobjekts (`DeviceItem`) vom aktuellen Gerät (`Device`).

Die folgende Tabelle beschreibt die **Navigatoren** des `OnlineDriveObjectContainer`:

Name	Datentyp	Beschreibung
<code>OnlineDriveObjects</code>	<code>OnlineDriveObjectComposition</code>	Gibt eine Liste mit den verfügbaren Online-Antriebsobjekten (<code>OnlineDriveObject</code> (Seite 1143)) zurück. Die Antriebsobjekte ermöglichen den Zugriff auf Antriebsparameter.
<code>DriveDataEncryption</code>	<code>Siemens.Engineering.MC.Drives.SecurityObjects.DriveDataEncryption</code>	Konfiguriert die Verschlüsselung der Antriebsdaten (DDE, Drive Data Encryption) für SINAMICS-Antriebe ab Firmware-Version V6.1.

5.23.4.20 StartDriveDownloadCheckConfiguration

StartDriveDownloadCheckConfiguration

Die Klasse `StartDriveDownloadCheckConfiguration` ist abgeleitet von der Klasse `DownloadCheckConfiguration` und hat die gleichen Eigenschaften. Die Klasse `DownloadCheckConfiguration` ist in der Standard-Openness-Hilfe beschrieben.

Diese Klasse stellt die Konfigurationseinstellungen über Optionskästchen vom Anwender zur Verfügung.

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
<code>Checked</code>	<code>bool</code>	Gibt die aktuelle Einstellung der Konfiguration zurück oder aktiviert/deaktiviert die Konfiguration.

Siehe auch

[Download](#) (Seite 1157)

5.23.4.21 SafetyTelegram

SafetyTelegram

Die Klasse `SafetyTelegram` steht für ein Telegramm des Antriebsobjekts. Bei Fehlern in den Schreib-Attributen erscheint die Exception `EngineeringTargetInvocationException`.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
TelegramNumber	Int32	Nummer der Haupttelegramms. Freie Telegramme werden mit dem Wert 999 gekennzeichnet.
Type	TelegramType (Seite 1150)	Telegrammtyp der als Enum "TelegramType" zurückgegeben wird.
Addresses	AddressComposition (Seite 1131)	Zusammenstellung aller Adressen eines Telegramms.
PKW	Telegram (Seite 1148)	Zusammenstellung aller PKW-Kanäle des Telegramms. null, wenn das Telegramm keinen PKW-Teil hat.

5.23.4.22 TechnologyExtension

TechnologyExtension

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriff	Beschreibung
Identifier	string	ReadOnly	Gibt den Identifier der Technology Extension zurück.
Name	string	ReadOnly	Gibt den Namen der Technology Extension zurück.
DisplayName	string	ReadOnly	Gibt den DisplayName der Technology Extension zurück.
IsActive	boolean	ReadOnly	Gibt den Aktivierungsstatus der Technology Extension zurück.

Die folgende Tabelle beschreibt die **Navigatoren** der Klasse:

Name	Typ	Zugriff	Beschreibung
Parent	IEngineeringObject	ReadOnly	Engineering Object Model-Eltern-element dieses Objekts.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Rückgabewert	Parameter	Beschreibung
Activate	bool	-	Aktiviert die Technology Extension.
Deactivate	bool	-	Deaktiviert die Technology Extension.

Siehe auch

Technology Extensions verwenden (Seite 1188)

5.23.4.23 TechnologyExtensionComposition

TechnologyExtensionComposition

Die Klasse `TechnologyExtensionComposition` enthält alle Technology Extensions, die einem bestimmten DriveObject zugewiesen sind. Alle Technology Extensions werden gleich behandelt.

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriff	Beschreibung
Count	int	ReadOnly	Gibt die Anzahl der Elemente zurück, die in <code>TechnologyExtensionComposition</code> enthalten sind.

Die folgende Tabelle beschreibt die **Navigatoren** der Klasse:

Name	Typ	Zugriff	Beschreibung
Parent	<code>IEngineeringObject</code>	ReadOnly	Engineering Object Model-Eltern-element dieses Objekts.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Rückgabewert	Parameter	Beschreibung
Find	<code>TechnologyExtensionItem</code>	string name/ identifier	Findet eine Technology Extension anhand des Namens/Identifiers.
Contains	bool	<code>TechnologyExtensionItem</code>	Ermittelt, ob ein Element in <code>TechnologyExtensionComposition</code> enthalten ist.
GetEnumerator	<code>IEnumerator<TechnologyExtension></code>	-	Gibt einen Enumerator zurück, der eine Auflistung der verfügbaren Technology Extensions durchläuft.

5.23.4.24 TechnologyExtensionInstallationProvider

TechnologyExtensionInstallationProvider

Die Klasse `TechnologyExtensionInstallationProvider` ist eine Funktion des TIA Portals und ermöglicht die Installation von `TechnologyExtensionPackage`.

Die folgende Tabelle beschreibt die **Navigatoren** der Klasse:

Name	Typ	Zugriff	Beschreibung
<code>TechnologyExtensionPackages</code>	<code>TechnologyExtensionPackageComposition</code>	ReadOnly	Zusammenstellung von <code>TechnologyExtensionPackages</code> .

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Typ des Rückgabewerts	Parameter	Beschreibung	Ausnahmen
Install	bool	ReadOnly	Diese Methode installiert ein <code>TechnologyExtensionPackage</code> . Parameterbeschreibung filePath: Vollqualifizierter Pfadname der zu installierenden .tec-Datei.	EngineeringException
Uninstall	bool	ReadOnly	Diese Methode deinstalliert ein <code>TechnologyExtensionPackage</code> . Parameterbeschreibung packagelIdentifier: Identifier, um das zur Deinstallation vorgesehene <code>TechnologyExtensionPackage</code> zu ermitteln. confirmUninstallInUse: Mit dieser Option bestätigen Sie die Deinstallation einer Technology Extension, die im geöffneten Projekt verwendet wird.	EngineeringException

5.23.4.25 TechnologyExtensionPackage

TechnologyExtensionPackage

Die Klasse `TechnologyExtensionPackage` ermöglicht den Zugriff auf Technology Extensions.

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Zugriff	Beschreibung
Identifier	string	ReadOnly	Gibt den Identifier des <code>TechnologyExtensionPackage</code> zurück. Die Syntax zur Definition lautet: <name>/<version>
Name	string	ReadOnly	Gibt den Namen des <code>TechnologyExtensionPackage</code> zurück.
DisplayName	string	ReadOnly	Gibt den DisplayName des <code>TechnologyExtensionPackage</code> zurück.

Die folgende Tabelle beschreibt die **Navigatoren** der Klasse:

Name	Typ	Zugriff	Beschreibung
Parent	IEngineeringObject	ReadOnly	Engineering Object Model-Eltern-element dieses Objekts.

5.23.4.26 Telegram

Telegram

Die Klasse `Telegram` ermöglicht den Zugriff auf den Aufbau eines Telegramms von einem Antriebsobjekt.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public sealed class Telegram
```

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
TelegramNumber	Int32	Gibt die Nummer des Haupttelegramms zurück oder legt die Nummer fest. Ein freies Telegramm hat die Nummer 999.
Type	TelegramType (Seite 1150)	Gibt den Typ des Telegramms als Enum <code>TelegramType</code> zurück.
Addresses	AddressComposition (Seite 1131)	Gibt eine <code>AddressComposition</code> mit Informationen zur Adresse zurück.
PKW	Telegram	Gibt den Kanal PKW als <code>Telegram</code> zurück. <code>null</code> , wenn die Eigenschaft nicht verfügbar ist Ein Telegramm mit PKW ist z. B. Telegramm 353.
HardwareIdentifier	IEnumerable<IBrowsable>	Gibt Liste der Hardware Identifier zurück.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
<code>CanChangeTelegram(Int32)</code>	Gibt <code>true</code> zurück, wenn das Telegramm in den parametrisierten Standardtyp geändert werden kann.
<code>GetSize(AddressIoType (Seite 1132))</code>	Gibt die Größe der Eingänge oder Ausgänge des Telegramms zurück.

Name	Beschreibung
CanChangeSize (AddressloType (Seite 1132), Int32, bool)	Gibt <code>true</code> zurück, wenn die Größe des Telegramms wie parametrisiert geändert werden kann. Standardtelegramme können nur vergrößert werden. Das Beibehalten der bisherigen Telegrammadresse wird berücksichtigt, wenn die Option mit <code>true</code> parametrisiert ist.
ChangeSize (AddressloType (Seite 1132), Int32, bool)	Gibt <code>true</code> zurück, wenn die Größe des Telegramms wie parametrisiert geändert werden konnte. Das Beibehalten der bisherigen Telegrammadresse wird berücksichtigt, wenn die Option mit <code>true</code> parametrisiert ist.

5.23.4.27 TelegramComposition

TelegramComposition

Die Klasse `TelegramComposition` ermöglicht den Zugriff auf die Telegramme eines Antriebsobjekts. Der Aufbau eines Telegramms kann über die Klasse `Telegram` (Seite 1148) ausgelesen werden.

Beachten Sie, dass durch die Klasse `TelegramComposition` referenzierte Objekte ungültig werden können. Z. B. wird nach Änderung der Telegrammgröße das Objekt `Telegram` (Seite 1148) ungültig.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public sealed class TelegramComposition
```

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Beschreibung
<code>CanInsertAdditionalTelegram(Int32, Int32)</code>	Gibt <code>true</code> zurück, wenn eine Erweiterung entsprechend den parametrisierten Größen (Eingangs- und Ausgangsgröße) erstellt werden kann.
<code>CanInsertTorqueTelegram(Int32, telegramNumber)</code>	Prüft beim Hinzufügen eines neuen Torque-Telegramms mit einer bereits vorhandenen Telegramm-Nummer, ob dies möglich ist.
<code>CanInsertSupplementaryTelegram(Int32)</code>	Gibt <code>true</code> zurück, wenn ein Zusatztelegramm entsprechend der parametrisierten Telegrammnummer erstellt werden kann.

Name	Beschreibung
<code>EraseTelegram(TelegramType)</code>	Löscht ein Telegramm mit bekanntem Telegrammtyp aus dem Antriebsobjekt. Gibt <code>true</code> zurück, wenn das parametrisierte Telegramm gelöscht werden konnte. Wird ein Torque-Telegramm als Typ verwendet, so wird das Objekt "torqueTelegram" gelöscht. Wird ein Safety Integrated-Telegramm als Typ verwendet, so wird das Objekt "safetyTelegram" gelöscht. Standardtelegramme können nicht gelöscht werden. Im Fehlerfall wird eine <code>EngineeringTargetInvocationException</code> ausgelöst.
<code>Find(TelegramType)</code>	Gibt das Objekt Telegram (Seite 1148) zurück, wenn es über den parametrisierten Telegrammtyp oder Safety Integrated-Telegrammtyp gefunden werden konnte. <code>null</code>, wenn das Telegramm nicht gefunden wird. Wird ein Torque-Telegramm als Typ verwendet, wird das Objekt <code>torqueTelegram</code> zurück gegeben, wenn es existiert. Wird ein Safety Integrated-Telegramm als Typ verwendet, wird das Objekt <code>safetyTelegram</code> zurück gegeben, wenn es existiert. Beispiel <pre>Telegram telegram = telegrams.Find(TelegramType.MainTelegram);</pre>
<code>InsertAdditionalTelegram(Int32, Int32)</code>	Erstellt für das Antriebsobjekt eine Erweiterung entsprechend den parametrisierten Größen und gibt <code>true</code> zurück, wenn die Erweiterung eingefügt werden konnte. Im Fehlerfall wird eine <code>EngineeringTargetInvocationException</code> ausgelöst.
<code>InsertTorqueTelegram(Int32, telegramNumber)</code>	Fügt einem Antriebsobjekt ein neues Torque-Telegramm mit einer bereits vorhandenen Telegramm-Nummer zu.
<code>InsertSafetyTelegram(Int32, telegramNumber)</code>	Fügt ein Safety Integrated-Telegramm mit seiner vorgegebenen Telegrammnummer einem Antriebsobjekt hinzu.
<code>InsertSupplementaryTelegram(Int32)</code>	Erstellt das Zusatztelegramm mit der parametrisierten Telegrammnummer und gibt <code>true</code> zurück, wenn das Telegramm eingefügt werden konnte. Im Fehlerfall wird eine <code>EngineeringTargetInvocationException</code> ausgelöst.

5.23.4.28 TelegramType

TelegramType

Das Enum `TelegramType` enthält vordefinierte Telegrammtypen.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Syntax** der Klasse:

```
public enum TelegramType
```

Die folgende Tabelle beschreibt die **Enum-Einträge**:

Name	Beschreibung
MainTelegram	ID des Haupttelegramms
SupplementaryTelegram	ID des Zusatztelegramms
AdditionalTelegram	ID einer Erweiterung

5.23.4.29 TorqueTelegram

TorqueTelegram

Die Klasse `TorqueTelegram` steht für ein Telegramm des Antriebsobjekts. Bei Fehlern in den Schreib-Attributen erscheint die Exception `EngineeringTargetInvocationException`.

Namespace: `Siemens.Engineering.MC.Drives`

Assembly: `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Die folgende Tabelle beschreibt die **Eigenschaften** der Klasse:

Name	Datentyp	Beschreibung
TelegramNumbe r	Int32	Nummer der Telegramms.
Type	TelegramType (Sei- te 1150)	Telegrammtyp der als Enum "TelegramType" zurückgegeben wird.
Addresses	AddressComposition (Seite 1131)	Zusammenstellung aller Adressen eines Telegramms.
PKW	Telegram (Sei- te 1148)	Zusammenstellung aller PKW-Kanäle des Telegramms. null, wenn das Telegramm keinen PKW-Teil hat.

5.23.4.30 AxisEncoderHardwareConnectionInterface

AxisEncoderHardwareConnectionInterface

Die Klasse ermöglicht den Zugriff auf Eigenschaften des Technologieobjekts.

Die folgende Tabelle beschreibt die **Methoden** der Klasse:

Name	Parameter	Rückgabewert	Beschreibung
Connect	Telegram	bool	Legt den Gebertyp des nicht-DRIVECLiQ-Gebers auf rotatorisch oder linear fest.
Connect	Telegram, ConnectOption	bool	Legt den Gebertyp des DRIVE-CLiQ-Gebers auf rotatorisch oder linear sowie absolut oder inkrementell fest. Nicht zur Verwendung mit nicht-DRIVE-CLiQ-Gebern vorgesehen. G220-Antriebe mit Firmware-Version V6.2 unterstützen nur den Gebertyp absolut oder inkrementell. Diese API kann zur Einstellung verwendet werden.

5.23.4.31 UmacConfiguration (ab SINAMICS FW V6.1)

Die Klasse UmacConfiguration ermöglicht, für SINAMICS-Antriebe ab Firmware-Version V6.1 die Funktion Benutzerverwaltung und Zugriffskontrolle (UMAC, User Management and Access Control) mit Openness zu aktivieren und zu deaktivieren.

Die folgende Tabelle beschreibt die **Methoden** der Klasse UmacConfiguration:

Name	Parameter	Rückgabewert	Beschreibung	Ausnahmen
Activate	-	bool	UmacConfiguration für den Antrieb aktivieren.	-
Deactivate	-	bool	UmacConfiguration für den Antrieb deaktivieren.	-

Hinweis

Klasse Security

Die Klasse Security ist eine übergeordnete Klasse. Die Klasse wird im Hintergrund verwendet, um auf alle APIs zuzugreifen, die mit Security zu tun haben.

5.23.4.32 UpdateChecksums (ab SINAMICS FW V6.1)

Die Klasse UpdateChecksums ermöglicht, für SINAMICS-Antriebe ab Firmware-Version V6.1 Safety Integrated Functions mit Openness zu konfigurieren.

Die folgende Tabelle beschreibt die **Methoden** der Klasse UpdateChecksums:

Name	Parameter	Rückgabewert	Beschreibung	Ausnahmen
UpdateChecksums	-	bool	Berechnet und überschreibt den Prüfsummenwert von Safety Integrated-Referenzparametern neu.	-

5.23.5 Codebeispiele

Die folgenden Codebeispiele beschreiben die grundsätzliche Vorgehensweise für verschiedene Anwendungsfälle. Der Code ist nicht zwingend vollständig und kompilierbar.

5.23.5.1 Aktivierungsstatus ermitteln

Die folgenden Beispiele zeigen, wie Sie den Aktivierungs-Status für S120-Antriebe offline oder online ermitteln können:

Aktivierungs-Status eines S120-Antriebs offline ermitteln

```
using Siemens.Engineering.MC.Drives;

DriveFunctionInterface dfi = ...
DriveObjectActivation driveObjectActivation =
dfi.DriveObjectFunctions.DriveObjectActivation;
//driveObjectActivation can be null in case of the actual driveobject does not support
activation.

//change activation state
driveObjectActivation.ChangeActivationState(DriveObjectActivationState.Deactivate);

//get the activation state
DriveObjectActivationState activationState = driveObjectActivation.ActivationState;

//get the Is Active property
bool isActive = driveObjectActivation.IsActive;
```

Aktivierungs-Status eines S120-Antriebs online ermitteln

```
using Siemens.Engineering.MC.Drives;

OnlineDriveFunctionInterface onlinedfi = ...
DriveObjectActivation driveObjectActivation = onlinedfi.DriveObjectActivation;
//driveObjectActivation can be null in case of the actual driveobject does not support
activation in online.

//change activation state
driveObjectActivation.ChangeActivationState(DriveObjectActivationState.Deactivate);

//get the activation state
DriveObjectActivationState activationState = driveObjectActivation.ActivationState;

//get the IsActive property
bool isActive = driveObjectActivation.IsActive;
```

5.23.5.2 Antriebsfunktionen durchführen

Die folgenden Beispiele zeigen, wie Sie auf einfache Weise offline oder online auf Antriebsfunktionen zugreifen können:

Antriebsfunktionen offline durchführen

```
using Siemens.Engineering.MC.Drives;
```

Antriebsfunktionen offline durchführen

```
DriveObject driveObject = ...
DriveFunctionInterface dfi = driveObject.GetService<DriveFunctionInterface>();

// dfi can be null in case of the actual driveobject does not support it.
```

Antriebsfunktionen online durchführen

```
using Siemens.Engineering.MC.Drives;
OnlineDriveObject onlineDriveObject = ...
OnlineDriveFunctionInterface onlineDfi =
onlineDriveObject.GetService<OnlineDriveFunctionInterface>();

// onlineDfi can be null in case of the actual onlineDriveObject
// does not support it or if the device is offline.
```

5.23.5.3 Antrieb anlegen

Mit der Methode `CreateWithItem()` der `Collection Devices` legen Sie einen Antrieb an.

Die Antriebe werden über Parameter der Methode spezifiziert. Das Format der Parameter ist nachfolgend beschrieben.

Antrieb G120 anlegen

Format der Parameter für G120:

```
CreateWithItem(@"OrderNumber: mlfb / FirmwareVersion /",
"NameOfTheDevice", positionNumber)
```

Das folgende Beispiel zeigt, wie Sie einen G120-Antrieb anlegen.

Antrieb G120 anlegen

```
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
Project tiaproject= portal.Projects.Open("..."); //The path of the project

Device s120Device =
tiaproject.Devices.CreateWithItem(@"OrderNumber:6SL3246-0BA22-1FA0/4.7.6/"
, "Device_0", null);
```

Antrieb S120, S150, S200, S210, MV, G130, G150 und G220 anlegen

Format der Parameter für S120, S150, S200, S210, MV, G130, G150 und G220-Antriebe:

```
CreateWithItem(@"OrderNumber: mlfb / FirmwareVersion /
AdditionalTypeIdentifier", "NameOfTheDevice", positionNumber)
```

Mögliche Werte für `AdditionalTypeIdentifier`:

- Leerstring (Z. B. für G120)
- S120
- S150

- S200
- S210
- MV
- G130
- G150
- G220

Das folgende Beispiel zeigt, wie Sie einen S120-Antrieb anlegen.

Antrieb S120 anlegen

```
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
Project tiaproject= portal.Projects.Open("..."); //The path of the project

Device s120Device =
tiaproject.Devices.CreateWithItem(@"OrderNumber:6SL3040-1MA01-0Axx/V4.8/
S120", "Device_0", null);
```

5.23.5.4 Antriebskomponente anlegen

Mit der Methode `PlugNew()` eines `Device` Objekts legen Sie eine Antriebskomponente für ein Antriebsgerät an.

Das folgende Beispiel zeigt, wie Sie eine Antriebskomponente anlegen.

Motormodul anlegen

```
DeviceItem subModul = sdrDevice.PlugNew(@"OrderNumber:6SL3xxx-xxxxxx-xxxx",
"MotorModul", 65535);
```

5.23.5.5 Antriebsobjekt ermitteln

Die folgenden Beispiele zeigen, wie Sie Antriebsobjekte offline und online ermitteln.

Offline-Antriebsobjekt ermitteln

```
using Siemens.Engineering.MC.Drives;
//G devices
Project project = portal.Projects.Open("..."); //Destination folder to
open the project
DeviceItem item = project.Devices[0].Items[0].Items[0];
DriveObject driveObject =
item.GetService<DriveObjectContainer>().DriveObjects[0];

//S devices
Project project = portal.Projects.Open("..."); //Destination folder to
open the project
DeviceItem item = project.Devices[0].Items[0];
DriveObject driveObject =
item.GetService<DriveObjectContainer>().DriveObjects[0];
```

Online-Antriebsobjekt ermitteln

```
using Siemens.Engineering.MC.Drives;
//G devices
Project project = portal.Projects.Open("..."); //Destination folder to
open the project
DeviceItem item = project.Devices[0].Items[0].Items[0];
OnlineDriveObject onlineDriveObject =
item.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];

//S devices
Project project = portal.Projects.Open("..."); //Destination folder to
open the project
DeviceItem item = project.Devices[0].Items[0];
OnlineDriveObject onlineDriveObject =
item.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];
```

5.23.5.6 Antriebsobjekt-Typ ermitteln

Das folgende Beispiel zeigt, wie Sie den aktuellen Antriebsobjekt-Typ und die alternativ einstellbaren Typen ermitteln können.

Antriebsobjekt-Typen ermitteln

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
DriveObject driveObject = ...
DriveFunctionInterface dfi = driveObject.GetService<DriveFunctionInterface>();
DriveObjectTypeHandler driveObjectTypeHandler =
dfi.DriveObjectFunctions.DriveObjectTypeHandler;

// dfi can be null in case of the actual driveobject does not support it.

// driveObjectTypeHandler can be null, if the actual driveObject does not support it.

// Get the possible drive object types on the drive object.
DriveObjectTypeComposition possibleDriveObjectTypes =
driveObjectTypeHandler.PossibleDriveObjectTypes;

// Get the current drive object type on the drive object.
DriveObjectType currentDriveObjectType = driveObjectTypeHandler.CurrentDriveObjectType;

//Call the ChangeDriveObjectType method with the current drive object type.
//The method parameter should be the target drive object type.
driveObjectTypeHandler.ChangeDriveObjectType(possibleDriveObjectTypes[0]);
```

5.23.5.7 BICO-Parameter lesen und schreiben

Das folgende Beispiel zeigt, wie Sie Werte von BICO-Parametern lesen und schreiben. Für den Zugriff benötigen Sie ein Antriebsobjekt.

BICO-Parameter lesen

```
using Siemens.Engineering.MC.Drives;
```


BICO-Parameter lesen

```

DriveParameter bicoSink= driveObject.Parameters.Find("p681");
if(bicoSink!=null)
{
    if(bicoSink.Value is DriveParameter)
    {
        DriveParameter bicoSourceValue = bicoSink.Value as DriveParameter;
        Console.WriteLine("The value of parameter " + bicoSink.Name + ": " +
bicoSource.Name + " " + bicoSource.ParameterText);
    }
    else if (bicoSink.Value == null)
    {
        Console.WriteLine("Value contains an invalid connection or the source
parameter is not accessible via Openness");
    }
    else
    {
        Console.WriteLine("The value of parameter " + bicoSink.Name + ": " +
bicoSink.Value.ToString());
    }
}

```

BICO-Parameter schreiben

```

using Siemens.Engineering.MC.Drives;
DriveParameter bicoSource= driveObject.Parameters.Find("r19");
DriveParameter bicoSink = driveObject.Parameters.Find("p738");
if(bicoSource != null)
{
    try
    {
        bicoSink.Value = bicoSource;
    }
    catch(UserException ex)
    {
        Console.WriteLine("Write failure :" + ex.Message);
    }
}

```

5.23.5.8 Download

Nach dem Start des Download müssen Sie die Konfigurationseinstellungen anpassen bzw. bestätigen. Die Konfigurationseinstellungen werden als Kindobjekte vom Objekt `DownloadConfiguration` zur Verfügung gestellt und gibt es in drei verschiedenen Typen:

- `StartDriveDownloadCheckConfiguration`
- `DownloadSelectionConfiguration`
- `DownloadPasswordConfiguration`

Das folgende Beispiele zeigt die Auswertung der verschiedenen Typen von Konfigurationseinstellungen im PreDownload Delegate.

Auswertung der Konfigurationseinstellungen nach Start des Download

```
using Siemens.Engineering.Download;
using Siemens.Engineering.Online;

static void PreDownload(DownloadConfiguration configuration)
{
    Console.WriteLine(configuration.Message);
    StartDriveDownloadCheckConfiguration sdcc = configuration as
    StartDriveDownloadCheckConfiguration;
    if (sdcc != null)
    {
        sdcc.Checked = true;
        return;
    }

    DownloadPasswordConfiguration downloadPasswordConfiguration =
    configuration as DownloadPasswordConfiguration;

    if (downloadPasswordConfiguration != null)
    {
        SecureString s = new SecureString();
        string passwordText = "password";
        foreach (var str in passwordText)
        {
            s.AppendChar(str);
        }

        downloadPasswordConfiguration.SetPassword(s);
        return;
    }

    DownloadSelectionConfiguration downloadSelectionConfiguration =
    configuration as DownloadSelectionConfiguration;

    if (downloadSelectionConfiguration != null)
    {
        downloadSelectionConfiguration.SelectedIndex = 0;
        return;
    }
}
```

Das folgende Beispiele zeigt, wie Sie ein Projekt in das Gerät laden.

Download in das Gerät S120

```
using Siemens.Engineering.Download;
using Siemens.Engineering.Online;
```

Download in das Gerät S120

```
try
{
    DeviceItem item = ... //device item of the CU (e.g. :
project.Devices[0].Items[0].Items[0])
    DownloadProvider downloadProvider = item.GetService<DownloadProvider>();

    DownloadConfigurationDelegate pre = PreDownload;
    DownloadConfigurationDelegate post = PostDownload;

    ConnectionConfiguration connConfiguration =
downloadProvider.Configuration;
    ConfigurationMode configurationMode = connConfiguration.Modes.Find("PN/
IE");
    ConfigurationPcInterface pcInterface = configurationMode.PcInterfaces[0];
    ConfigurationSubnet subnet = pcInterface.Subnets.Find(/*subnet name*/);
    IConfiguration configuration = subnet.Addresses.Find(/*IP address of the
device*/);
    downloadProvider.Download(configuration, pre, post,
DownloadOptions.Software);
}
catch (Exception e)
{
    Console.WriteLine(e.Message);
}

//configuration handling before download
static void PreDownload(DownloadConfiguration configuration)
{
    Console.WriteLine(configuration.Message);
    StartDriveDownloadCheckConfiguration dcc = configuration as
StartDriveDownloadCheckConfiguration ;
    if (dcc != null)
    {
        dcc.Checked = true;
    }
}

//configuration handling after download
static void PostDownload(DownloadConfiguration configuration)
{
    Console.WriteLine(configuration.Message);
}
```

Zusätzliche Einstellungen für Downloads bei Antrieben ab SINAMICS FW V6.1

Wenn Sie einen Antrieb mit SINAMICS FW V6.1 oder höher einem Projekt hinzufügen, sind die Security-Einstellungen für diesen Antrieb in der Werkseinstellung.

Ein Download von so einem Antrieb ist nur möglich, wenn für das Projekt UMAC aktiviert wird.

UMAC konfigurieren

```
UmacConfigurator UmacConfiguratorService =  
project.GetService<UmacConfigurator>();var projectUsers =  
UmacConfiguratorService.ProjectUsers;
```

Einen Benutzer für das Projekt erstellen

```
ProjectUser projectUser = projectUsers.Create("AdminUser",  
"AdminUser123#".ToSecureString());
```

Eine System-Rolle hinzufügen

```
projectUser.Roles.Add(systemRole);
```

Zusätzlich erfordert ein Download ein gültiges TLS-Zertifikat wie im folgenden Beispiel beschrieben:

TLS-Zertifikat einstellen

```
private static OnlineConfigurationDelegate m_OnlineConfigurationDelegate =  
null;  
private bool TryDownloadToDeviceTest()  
{  
    String newAddress = ...// network ip;  
    try  
    {  
        Device device =...//device  
        DeviceItem item = device.Items[0].Items[0];  
        DownloadProvider downloadProvider = item.GetService<DownloadProvider>();  
        ConnectionConfiguration conf = downloadProvider.Configuration;  
    }  
}
```

Benachrichtigung für TLS-Ereignisse aktivieren

```
if (m_OnlineConfigurationDelegate == null)  
{  
    m_OnlineConfigurationDelegate = OnlineCallBackMethod;  
    conf.OnlineLegitimation += m_OnlineConfigurationDelegate;  
}
```

Konfigurationseinstellungen für Download einstellen

```
//perform download
{
catch (Exception e)
{
    Console.WriteLine(e.ToString());
    return false;
}
return true;
}

private static void OnlineCallBackMethod(OnlineConfiguration
onlineConfiguration)
{
    TlsVerificationConfiguration verificationConfiguration =
onlineConfiguration as TlsVerificationConfiguration;
    if (verificationConfiguration != null)
    {
        verificationConfiguration.CurrentSelection =
TlsVerificationConfigurationSelection.Trusted;
    }
}
```

Siehe auch

Verschlüsselte Kommunikation mit SecureString-Passwörtern (Seite 1129)

5.23.5.9 DRIVE-CLiQ-Verbindungen bearbeiten

Das folgende Beispiel zeigt, wie Sie mit Openness DRIVE-CLiQ-Verbindungen bearbeiten können.

DRIVE-CLiQ-Verbindungen bearbeiten

```
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
```

5.23 Funktionen für Startdrive

DRIVE-CLiQ-Verbindungen bearbeiten

```

Project tiaproject = portal.Projects.Open(new
FileInfo(@"C:\Users\testUser\Documents\Automation\Project109\Project109.ap
15")); //The path of the project

Device device =
tiaproject.Devices.CreateWithItem(@"OrderNumber:6SL3040-1MA01-0Axx/V4.8/
S120", "Device_0", null); //Create the device

DeviceItem subModul = device.PlugNew(@"OrderNumber:6SL3x2x-1xxxx-xxxx",
"", 65535); //Plug a submodul (in this case : Motor modul)

DeviceItem cu = device.DeviceItems[1];           // The CU is always the 1
indexed in the DeviceItems of the Device

DeviceItem subModulDQInterface = subModul.DeviceItems[0];           //We need
the DQ interface from the submodul
DeviceItem cuDQInterface = cu.DeviceItems[3];           //This is the
DQ interface of the CU

NetworkPort subModulDQ =
((IEngineeringServiceProvider)subModulDQInterface.DeviceItems[0]).GetServi
ce<NetworkPort>(); //We need the DriveCliq port of the DQ interface from
the submodul
NetworkPort cuDQ =
((IEngineeringServiceProvider)cuDQInterface.DeviceItems[0]).GetService<Net
workPort>();           //We need the DriveCliq port of the DQ interface
from the CU

cuDQ.DisconnectFromPort(subModulDQ); //Delete the connection between the
two ports (automatically created when plugging a modul)
cuDQ.ConnectToPort(subModulDQ); //Create a new connection

```

5.23.5.10 Erste Schritte in Startdrive durchführen

Das folgende Beispiel zeigt, wie Sie einen laufenden TIA-Portal-Prozess lokalisieren oder erzeugen können.

TIA Portal-Prozess finden oder erzeugen

```

using Siemens.Engineering;

// Get the list of the running TIA Portal processes.
IList<TiaPortalProcess> procs = TiaPortal.GetProcesses();
TiaPortal portal;
// When there is at least one running TIA Portal, we will attach to the first from the list.
if (procs.Count != 0)
{
    portal = procs[0].Attach();
}
// When there is no running TIA Portal, we create one.
else
{
    portal = new TiaPortal(TiaPortalMode.WithUserInterface);
}

```

Das folgende Beispiel zeigt, wie Sie ein Startdrive-Projekt lokalisieren oder erzeugen können.

Startdrive-Projekt finden oder erzeugen

```
using Siemens.Engineering;
Project project;
// When the portal has one project, we save it in a variable.
if (portal.Projects.Count == 1)
{
    project = portal.Projects[0];
}
// When there is no existing project, we create one with a specific path, and the actual time
else
{
    project = portal.Projects.Create(
        new DirectoryInfo(@"C:\Projects\Project_" + DateTime.Now.Ticks),
        DateTime.Now.Ticks.ToString());
}
```

Das folgende Beispiel zeigt, wie Sie ermitteln können, ob eine bestimmte Startdrive-Variante (Package und Version) installiert ist.

Ermitteln, ob die gewünschte Startdrive-Version installiert ist

```
using Siemens.Engineering;
if (tiaProcess.InstalledSoftware.Any(sw => sw.Name.Equals("SINAMICS Startdrive Advanced")
    && sw.Version.Equals("V15"))) { Console.WriteLine("Startdrive is available");}
// "V15" is the current startdrive version started at December 2017.
// "V15.1" will be the current startdrive version beginning with December 2018.
// "SINAMICS Startdrive Basic" and "SINAMICS Startdrive Advanced" are the 2 possible
startdrive function packages.
```

5.23.5.11 Geber-Typ festlegen

Das folgende Beispiel zeigt, wie Sie offline den Geber-Typ bzw. die Geber-Datensatznummer einstellen können.

Geber-Typ bzw. Geber-Datensatznummer offline über Hardware-Projektierung einstellen

```
using Siemens.Engineering.MC.Drives
using Siemens.Engineering.MC.Drives.DFI
using Siemens.Engineering.MC.Drives.Enums;
```

Geber-Typ bzw. Geber-Datensatznummer offline über Hardware-Projektierung einstellen

```

DeviceItem cuDeviceItem = m_Device.DeviceItems[1];
DriveObject cuDriveObject =
cuDeviceItem.GetService<DriveObjectContainer>().DriveObjects[0];

DriveFunctionInterface cuDriveFunctionInterface =
cuDriveObject.GetService<DriveFunctionInterface>();
HardwareProjection hardwareProjection = cuDriveFunctionInterface.HardwareProjection;

// To enable the setting of an EncoderType, the value of p96 should
// be set to 0 (Application class == [0] Expert) if it is present on
// the current G drive.
DriveParameter p96 = cuDriveObject.Parameters.Find("p96");
if (p96 != null)
{
    p96.Value = 0;
}

// Setting Encoder 1 on the drive.
// In case of encoders, we have to set several enums to define an
// encoder type. These enums are: EncoderInterface, EncoderType,
// AbsoluteIncrementalFlag, and RotaryLinearFlag.

// There can be a problem, if the given enum combination is not valid.
// In that case, it has to give back a feedback.

hardwareProjection.SetEncoder(
    EncoderInterface.Terminal,
    EncoderType.HTLTTL,
    AbsoluteIncrementalFlag.Incremental,
    RotaryLinearFlag.Rotary,
    1);

// It is possible to set 2 encoders to a motor, one with encoderNumber == 1
// and the other with encoderNumber == 2

```

Das folgende Beispiel zeigt, wie Sie mit einer Onlineverbindung den Geber-Typ bzw. die Geber-Datensatznummer einstellen können.

Geber-Typ bzw. Geber-Datensatznummer online über Hardware-Projektierung einstellen

```

using Siemens.Engineering.MC.Drives
using Siemens.Engineering.MC.Drives.DFI
using Siemens.Engineering.MC.Drives.Enums;

```


Geber-Typ bzw. Geber-Datensatznummer online über Hardware-Projektierung einstellen

```
DeviceItem cuDeviceItem = m_Device.DeviceItems[1];
OnlineDriveObject cuOnlineDriveObject =
cuDeviceItem.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];
OnlineDriveFunctionInterface cuDriveFunctionInterface =
cuOnlineDriveObject.GetService<OnlineDriveFunctionInterface>();
HardwareProjection hardwareProjection = cuDriveFunctionInterface.HardwareProjection;

// To enable the setting of an Encoder, the value of p96 should
// be set to 0 (Application class == [0] Expert) if it is present
// the current G drive.
DriveParameter p96 = cuOnlineDriveObject.Parameters.Find("p96");
if (p96 != null)
{
    p96.Value = 0;
}

// Setting Encoder 1 on the drive.
// In case of encoders we have to set several enums to define an
// encoder type. These enums are: EncoderInterface, EncoderType,
// AbsoluteIncrementalFlag, and RotaryLinearFlag.

// There can be a problem, if the given enum combination is not valid.
// In that case, it has to give back a feedback.

hardwareProjection.SetEncoder(
    EncoderInterface.Terminal,
    EncoderType.HTLTTL,
    AbsoluteIncrementalFlag.Incremental,
    RotaryLinearFlag.Rotary,
    1);

// It is possible to set 2 encoders to a motor, one with encoderNumber == 1
// and the other with encoderNumber == 2
```

Das folgende Beispiel zeigt, wie Sie die aktuelle Konfiguration des Gebers aus dem Antrieb auslesen können.

Konfiguration des Gebers auslesen

```
DeviceItem cuDeviceItem = m_Device.DeviceItems[1];
DriveObject cuDriveObject =
cuDeviceItem.GetService<DriveObjectContainer>().DriveObjects[0];
DriveFunctionInterface cuDriveFunctionInterface =
cuDriveObject.GetService<DriveFunctionInterface>();

HardwareProjection encoderProjection = cuDriveFunctionInterface.HardwareProjection;

// Before setting an encoder, p96 should be 0
// (See section "Projecting EncoderConfiguration")

encoderProjection.SetEncoder(
    EncoderInterface.Terminal,
    EncoderType.HTLTTL,
    AbsoluteIncrementalFlag.Incremental,
    RotaryLinearFlag.Rotary,
    1);

EncoderConfiguration encoderConfiguration =
encoderProjection.GetCurrentEncoderConfiguration(1);
```

Das folgende Beispiel zeigt, wie Sie die aktuelle Konfiguration des Gebers offline projektieren können.

Geber offline konfigurieren

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;
```

Geber offline konfigurieren

```
// Project encoder configuration in Offline state
DeviceItem cuDeviceItem = m_Device.DeviceItems[1];

DriveObject cuDriveObject =
cuDeviceItem.GetService<DriveObjectContainer>().DriveObjects[0];
DriveFunctionInterface cuDriveFunctionInterface =
cuDriveObject.GetService<DriveFunctionInterface>();
HardwareProjection hardwareProjection = cuDriveFunctionInterface.HardwareProjection;

// Before setting an encoder type, p96 should be 0
// (See section "Projecting EncoderConfiguration")

// To Project an EncoderConfiguration, first set e.g. Encoder 1 with .SetEncoder(...).
hardwareProjection.SetEncoder(
    EncoderInterface.Terminal,
    EncoderType.HTLTTL,
    AbsoluteIncrementalFlag.Incremental,
    RotaryLinearFlag.Rotary,
    1);

// Get the current configuration of Encoder 1.
EncoderConfiguration config = hardwareProjection.GetCurrentEncoderConfiguration(1);

// Fill out Encoder 1's configuration values.
config.RequiredConfigurationEntries.ToList().ForEach(ce =>
{
    switch (ce.Name)
    {
        case "p405.0":
            ce.Value = 0;
            break;
        case "p405.1":
            ce.Value = 1;
            break;
        case "p408":
            ce.Value = 1024;
            break;
        case "p425":
            ce.Value = 2048;
            break;
        default:
            break;
    }
});

// Project the Encoder 1 configuration to the device.
bool result =
cuDriveFunctionInterface.HardwareProjection.ProjectEncoderConfiguration(config, 1);
```

Das folgende Beispiel zeigt, wie Sie die aktuelle Konfiguration des Gebers online projektieren können.

Geber online konfigurieren

```
using Siemens.Engineering.MC.Drives;  
using Siemens.Engineering.MC.Drives.DFI;  
using Siemens.Engineering.MC.Drives.Enums;
```

Geber online konfigurieren

```
// Project encoder configuration in Online state
DeviceItem cuDeviceItem = m_Device.DeviceItems[1];
DeviceItem cuDeviceItemForOnline = m_Device.DeviceItems[0];
//You need the rack for going online on G120 Device, which is .DeviceItems[0]

// GoOnline...
// To use these function in Online, you have to use the OnlineDriveFunctionInterface
OnlineDriveObject onlineDriveObject =
cuDeviceItem.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];
OnlineDriveFunctionInterface onlineDfi =
onlineDriveObject.GetService<OnlineDriveFunctionInterface>();
HardwareProjection hardwareProjection = onlineDfi.HardwareProjection;

// Before setting an encoder, p96 should be 0
// (See section "Projecting EncoderConfiguration")

// To Project an EncoderConfiguration, first set e.g. Encoder 1 with .SetEncoder(...).

hardwareProjection.SetEncoder(
    EncoderInterface.Terminal,
    EncoderType.HTLTTL,
    AbsoluteIncrementalFlag.Incremental,
    RotaryLinearFlag.Rotary,
    1);

// Get the current configuration of Encoder 1.
EncoderConfiguration config = hardwareProjection.GetCurrentEncoderConfiguration(1);

// Fill out Encoder 1's configuration values.
config.RequiredConfigurationEntries.ToList().ForEach(ce =>
{
    switch (ce.Name)
    {
        case "p405.0":
            ce.Value = 0;
            break;
        case "p405.1":
            ce.Value = 1;
            break;
        case "p408":
            ce.Value = 1024;
            break;
        case "p425":
            ce.Value = 2048;
            break;
        default:
            break;
    }
});

// Project the Encoder 1 configuration to the device.
bool result = onlineDfi.HardwareProjection.ProjectEncoderConfiguration(config, 1);
```

Das folgende Beispiel zeigt, wie Sie den Konfigurationseingang in die Konsole schreiben können.

Konfigurationseingang heraus schreiben

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
MotorConfiguration motorConfig = hardwareProjection.GetCurrentMotorConfiguration(0);

foreach (var configurationEntry in motorConfig.RequiredConfigurationEntries)
{
    Console.WriteLine(configurationEntry.Name);
}
EncoderConfiguration encConfig = hardwareProjection.GetCurrentEncoderConfiguration(1);

foreach (var configurationEntry in encConfig.RequiredConfigurationEntries)
{
    Console.WriteLine(configurationEntry.Name);
}
```

Das folgende Beispiel zeigt, wie Sie DRIVE-CLiQ-Geber konfigurieren können.

DRIVE-CLiQ-Geber konfigurieren

```
using Siemens.Engineering.MC.Drives.Enums;
```

DRIVE-CLiQ-Geber konfigurieren

```
private bool TestEncoderProjectionForAcxDrives(){
    bool result = false;
    //Relevant drive already added to project m_Project.Devices[0]
    DeviceItem motorModule = m_Project.Devices[0].DeviceItems[2]; //get the motor module or
power module
    DriveObject mmDriveObject =
motorModule.GetService<DriveObjectContainer>().DriveObjects[0];
    DriveFunctionInterface mmDriveFunctionInterface =
mmDriveObject.GetService<DriveFunctionInterface>();
    HardwareProjection hwProjection = mmDriveFunctionInterface.HardwareProjection;

    try
    {
        EncoderConfiguration encoderConfiguration =
hwProjection.GetCurrentEncoderConfiguration(0);
        //get EncoderConfiguration for encoder at index 0

        IList<ConfigurationEntry> encoderTypes =
encoderConfiguration.EncoderTypes.ToList();
        //get the type of encoder, this would return only the applicable parameters p404.0
& p404.1

        bool setEncoderTypeResult =
encoderConfiguration.SetEncoderType(RotaryLinearFlag.Linear,
AbsoluteIncrementalFlag.Incremental);
        //SetEncoderType is optional and can be used to change Encoder Type.
        //SetEncoderType can be used to set Rotary or Linear as Encoder Type.
        //For DriveCliq encoder, along with Rotary/Linear, you can also set the Encoder type
as Incremental or Absolute.

        if (setEncoderTypeResult)
        {
            encoderConfiguration.RequiredConfigurationEntries.ToList().ForEach(e =>
            {
                if (e.Name.Equals("p404.0"))
                {
                    e.Value = true;
                }
            }); //set parameter p404.0 (Linear encoder) to true
            result = hwProjection.ProjectEncoderConfiguration(encoderConfiguration, 0);
        }
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message+ "\n" + e.StackTrace);
    }
    return result;
}
```

Das folgende Beispiel zeigt, wie Sie Nicht-Siemens-Geber ohne DRIVE-CLiQ konfigurieren können.

Nicht-Siemens-Geber ohne DRIVE-CLiQ konfigurieren

```
using Siemens.Engineering.MC.Drives.Enums;
```

Nicht-Siemens-Geber ohne DRIVE-CLiQ konfigurieren

```

private bool TestEncoderProjectionForAcxDrives()
{
    bool result = false;
    //Relevant drive already added to project m_Project.Devices[0]
    DeviceItem motorModule = m_Project.Devices[0].DeviceItems[2]; //get the motor module or
power module
    DriveObject mmDriveObject =
motorModule.GetService<DriveObjectContainer>().DriveObjects[0];
    DriveFunctionInterface mmDriveFunctionInterface =
mmDriveObject.GetService<DriveFunctionInterface>();
    HardwareProjection hwProjection = mmDriveFunctionInterface.HardwareProjection;

    try
    {
        EncoderConfiguration encoderConfiguration =
hwProjection.GetCurrentEncoderConfiguration(0);
        //get EncoderConfiguration for encoder at index 0

        IList<ConfigurationEntry> encoderTypes =
encoderConfiguration.EncoderTypes.ToList();
        //get the type of encoder, this would return only the applicable parameter p404.0

        bool setEncoderTypeResult =
encoderConfiguration.SetEncoderType(RotaryLinearFlag.Linear);
        //SetEncoderType is optional and can be used to change Encoder Type.
        //SetEncoderType can be used to set Rotary or Linear as Encoder Type.

        if (setEncoderTypeResult)
        {
            encoderConfiguration.RequiredConfigurationEntries.ToList().ForEach(e =>
            {
                if (e.Name.Equals("p404.0"))
                {
                    e.Value = true;
                }
            }); //set parameter p404.0 (Linear encoder) to true
            result = hwProjection.ProjectEncoderConfiguration(encoderConfiguration, 0);
        }
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message+ "\n" + e.StackTrace);
    }
    return result;
}

```


5.23.5.12 Geräte-Projektierung durchführen

Das folgende Beispiel zeigt, wie Sie für S120- und G120-Antriebe eine Geräte-Projektierung offline durchführen können.

Geräte-Projektierung offline bei S120- und G120-Antrieben durchführen

```
using Siemens.Engineering.MC.Drives
using Siemens.Engineering.MC.Drives.DFI
using Siemens.Engineering.MC.Drives.Enums;

DriveObject driveObject = ...
DriveFunctionInterface dfi = driveObject.GetService<DriveFunctionInterface>();
HardwareProjection hardwareProjection = dfi.HardwareProjection;

// dfi can be null in case of the actual driveobject does not support it.
// hardwareProjection can be null, if the actual driveObject does not support it.
// For example: On G120 drives, you have to use the CU as driveobject. On S120
// drives, you have to use the MotorModul as driveObject.
```

Das folgende Beispiel zeigt, wie Sie für S120- und G120-Antriebe eine Geräte-Projektierung online durchführen können.

Geräte-Projektierung online bei S120- und G120-Antrieben durchführen

```
using Siemens.Engineering.MC.Drives
using Siemens.Engineering.MC.Drives.DFI
using Siemens.Engineering.MC.Drives.Enums;

OnlineDriveObject onlineDriveObject = ...
OnlineDriveFunctionInterface onlineDfi =
onlineDriveObject.GetService<OnlineDriveFunctionInterface>();
HardwareProjection hardwareProjection = onlineDfi.HardwareProjection;

// onlineDfi can be null in case of the actual onlineDriveObject
// does not support it or if the device is offline.
// hardwareProjection can be null, if the actual onlineDriveObject
// does not support it.
// For example: On G120 drives, you have to use the CU as onlineDriveObject. On
// S120 drives, you have to use the MotorModul as onlineDriveObject.
```

5.23.5.13 Hardware Identifier ermitteln

Die folgenden Beispiele zeigen, wie Sie den Hardware Identifier (Seite 1148) verschiedener Objekte ermitteln.

Hardware Identifier von Telegramm ermitteln

```
using Siemens.Engineering.MC.Drives;

Device s120Drive = m_Project.Devices[1];
DeviceItem driveUnit = s120Drive.DeviceItems[0];
DriveObjectContainer driveObjectContainer =
driveUnit.GetService<DriveObjectContainer>();
DriveObject driveObject = driveObjectContainer.DriveObjects[0];
Telegram mainTelegram = driveObject.Telegrams[0];

IList<HwIdentifier> hwIdentifiers =
mainTelegram.HwIdentifiers?.ToList(); //get the Hardware Identifier
long identifier = hwIdentifiers[0].Identifier;
```

Hardware Identifier eines Module Access Point ermitteln

```

using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.HW.HwIdentifier;

Device s120Drive = m_Project.Devices[1];
DeviceItem driveUnit = s120Drive.DeviceItems[0];
DriveObjectContainer driveObjectContainer =
driveUnit.GetService<DriveObjectContainer>();
DriveObject driveObject = driveObjectContainer.DriveObjects[0];
Telegram mainTelegram = driveObject.Telegrams[0];

IList<HwIdentifier> hwIdentifiers =
mainTelegram.HwIdentifiers?.ToList(); //get the Hardware Identifier
long identifier = hwIdentifiers[0].Identifier;

Device deviceS120 = m_Project.Devices[1]; //S120 with Double Motor Module
Device deviceG120 = m_Project.Devices[2]; //G120
Device deviceG220 = m_Project.Devices[3]; //G220

ModuleAccessPoint moduleAccessPointS120 =
deviceS120.DeviceItems[1].GetService<ModuleAccessPoint>();
IList<HwIdentifier> hwIdentifiersS120 =
moduleAccessPointS120.HwIdentifiers.ToList();
long identifierS120 = hwIdentifiersS120[0].Identifier;

ModuleAccessPoint moduleAccessPointDriveAxis =
deviceS120.DeviceItems[2].GetService<ModuleAccessPoint>();
IList<HwIdentifier> hwIdentifiersDriveAxis =
moduleAccessPointDriveAxis.HwIdentifiers.ToList();
long identifierDriveAxis = hwIdentifiersDriveAxis[0].Identifier;

ModuleAccessPoint moduleAccessPointDriveAxisTwo =
deviceS120.DeviceItems[3].GetService<ModuleAccessPoint>();
IList<HwIdentifier> hwIdentifiersDriveAxisTwo =
moduleAccessPointDriveAxisTwo.HwIdentifiers.ToList();
long identifierDriveAxisTwo = hwIdentifiersDriveAxisTwo[0].Identifier;

ModuleAccessPoint moduleAccessPointG120 =
deviceG120.DeviceItems[1].GetService<ModuleAccessPoint>();
IList<HwIdentifier> hwIdentifiersG120 =
moduleAccessPointG120.HwIdentifiers.ToList();
long identifierG120 = hwIdentifiersG120[0].Identifier;

ModuleAccessPoint moduleAccessPointG220 =
deviceG220.DeviceItems[1].GetService<ModuleAccessPoint>();
IList<HwIdentifier> hwIdentifiersG220 =
moduleAccessPointG220.HwIdentifiers.ToList();
long identifierG220 = hwIdentifiersG220[0].Identifier;

```

5.23.5.14 Komponente für eine Antriebskomponente anlegen (nur S120)

Bei S120 haben Sie die Möglichkeit, eine Komponente unter einer Antriebskomponente anzulegen.

Für die Unterscheidung von Gebern geben Sie zusätzlich eine Typbezeichnung an. Die möglichen Typbezeichnungen und Einschränkungen bei Gebern (Encodern) sind in der Tabelle unten aufgelistet.

Das folgende Beispiel zeigt, wie Sie eine Komponente unter einer Antriebskomponente anlegen.

Motor und Geber unter einem Motormodul anlegen

```
DeviceItem subModul = sdrDevice.PlugNew(@"OrderNumber:6SL3xxx-xxxxx-xxxx",
"MotorModul", 65535);
```

```
//Plug a motor to the motor modul
subModul.Container.PlugNew(@"OrderNumber:1PH2092-4WG4x-xxxx",
"Motor_1", 65535);
```

```
//Plug an encoder to the motor modul
subModul.Container.PlugNew(@"OrderNumber:XExxxxx-xxxxx-xxxx//DRIVE-
CLIQ.202", "Encoder_1", 65535);
```

Typbezeichnungen für Geber und Einschränkungen

Beim Einfügen von Gebern über Openness gelten folgende Einschränkungen:

- Für manche Geber kann beim Einfügen über Openness nur ein unspezifisches Sensormodul angelegt. In diesen Fällen müssen Sie im TIA-Portal den konkreten Typ des Sensormoduls konfigurieren.
- Für ein Motormodul können maximal zwei Geber eingefügt werden.

Die folgende Tabelle listet die verfügbaren Typbezeichnungen für Geber auf.

DRIVE-CLiQ	Resolver	sin/cos	SSI	sin/cos+SSI	HTL/TTL	HTL/TTL+SSI	EnDat 2.1
DRIVE-CLIQ.202	Resolver.0	SIN_COS.0	SSI.0	SIN_COS+_SSI.0	HTL_TTL.0	HTL_TTL+SSI.0	EnDat_2.1.2051
DRIVE-CLIQ.204	Resolver.1001	SIN_COS.2001	SSI.3081	SIN_COS+_SSI.2081	HTL_TTL.3001	HTL_TTL+SSI.3088	EnDat_2.1.2052
DRIVE-CLIQ.212	Resolver.1002	SIN_COS.2002	SSI.3082	SIN_COS+_SSI.2082	HTL_TTL.3002	HTL_TTL+SSI.3090	EnDat_2.1.2053
DRIVE-CLIQ.214	Resolver.1003	SIN_COS.2003	SSI.9999	SIN_COS+_SSI.2083	HTL_TTL.3003	HTL_TTL+SSI.9999	EnDat_2.1.2054
DRIVE-CLIQ.242	Resolver.1004	SIN_COS.2004	-	SIN_COS+_SSI.2084	HTL_TTL.3005	-	EnDat_2.1.2055
DRIVE-CLIQ.244	Resolver.9999	SIN_COS.2005	-	SIN_COS+_SSI.9999	HTL_TTL.3006	-	EnDat_2.1.2151
DRIVE-CLIQ.9999	-	SIN_COS.2006	-	-	HTL_TTL.3007	-	EnDat_2.1.9999
DRIVE-CLIQ.10100	-	SIN_COS.2007	-	-	HTL_TTL.3008	-	EnDat_2.1.10100
-	-	SIN_COS.2008	-	-	HTL_TTL.3009	-	-
-	-	SIN_COS.2010	-	-	HTL_TTL.3011	-	-

DRIVE-CLiQ	Resolver	sin/cos	SSI	sin/cos+SSI	HTL/TTL	HTL/TTL+SSI	EnDat 2.1
-	-	SIN_COS.2012	-	-	HTL_TTL.3020	-	-
-	-	SIN_COS.2013	-	-	HTL_TTL.3109	-	-
-	-	SIN_COS.2110	-	-	HTL_TTL.9999	-	-
-	-	SIN_COS.2111	-	-	-	-	-
-	-	SIN_COS.2112	-	-	-	-	-
-	-	SIN_COS.9999	-	-	-	-	-

5.23.5.15 Motor-Typ und Motor-Konfiguration festlegen

Die folgenden Beispiele zeigen, wie Sie den Motortyp für G120-Antriebe über die Gerätekonfiguration einstellen können:

Motortyp offline über Hardware-Projektierung einstellen

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;
//Offline (Only on G120 drives)
DriveFunctionInterface dfi = ...
HardwareProjection hardwareProjection = dfi.HardwareProjection;

// hardwareProjection can be null in case of the actual driveObject
// does not support activation.
// Setting the required MotorType and DriveDataSetNumber on the drive.
// It is only supported on G120 drives.

//First parameter is the MotorType, Second parameter is the DriveDataSetNumber
hardwareProjection.SetMotorType(MotorType.InductionMotor, 0);

// There can be a problem, if the selected MotorType is not available
// on the drive. In that case, it has to give back a feedback.
```

Motortyp online über Hardware-Projektierung einstellen

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;
```

Motortyp online über Hardware-Projektierung einstellen

```
//Online (Only on G120 drives)
OnlineDriveFunctionInterface onlineDfi = ...
HardwareProjection hardwareProjection = onlineDfi.HardwareProjection;

// hardwareProjection can be null in case of the actual
// onlineDriveObject does not support activation.
// Setting the required MotorType and DriveDataSetNumber on
// the drive. It is only supported on G120 drives.

// First parameter is the MotorType, Second parameter is the DriveDataSetNumber
hardwareProjection.SetMotorType(MotorType.InductionMotor, 0);

// There can be a problem, if the selected MotorType is not available on
// the drive. In that case, it has to give back a feedback.
```

Die folgenden Beispiele zeigen, wie Sie den Motortyp für G120-Antriebe aus dem Antrieb auslesen können:

Motorkonfiguration offline ermitteln

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;

// Offline
// WARNING: You have to set the MotorType on G drives before
// you would like to get the current configuration, otherwise
// you will get an exception/feedback, which informs you that
// there is no motor set.
// On S120 drives, you don't have to do that, but you have to
// plug a motor to motor modul.( later on)

HardwareProjection hardwareProjection = dfi.HardwareProjection;
// Get the current motor configuration
// It needs a datasetnumber, which is currently only supported on G120 drives.
MotorConfiguration motorConfiguration = hardwareProjection.GetCurrentMotorConfiguration(0);
```

Motorkonfiguration online ermitteln

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;

// Online
// WARNING: You have to set the MotorType on G drives before
// you would like to get the current configuration, otherwise
// you will get an exception/feedback, which informs you that
// there is no motor set.

HardwareProjection hardwareProjection = onlineDfi.HardwareProjection;
// Get the current motor configuration
// It needs a datasetnumber, which is currently only supported on G120 drives.
MotorConfiguration motorConfiguration = hardwareProjection.GetCurrentMotorConfiguration(0);
```

Das folgende Beispiel zeigt, wie Sie eine Motorkonfiguration für G120-Antriebe durchführen können:

Motorkonfiguration für G120-Antriebe projektieren

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;
DeviceItem cuDeviceItem = m_Device.DeviceItems[1];
DriveObject cuDriveObject =
cuDeviceItem.GetService<DriveObjectContainer>().DriveObjects[0];
DriveFunctionInterface cuDriveFunctionInterface =
cuDriveObject.GetService<DriveFunctionInterface>();

HardwareProjection hardwareProjection = cuDriveFunctionInterface.HardwareProjection;
hardwareProjection.SetMotorType(MotorType.InductionMotor, 0);

MotorConfiguration config = hardwareProjection.GetCurrentMotorConfiguration(0);

config.RequiredConfigurationEntries.ToList().ForEach(ce =>
{
    switch (ce.Number)
    {
        case 305:
            ce.Value = 20;
            break;
        case 307:
            ce.Value = 30;
            break;
        case 311:
            ce.Value = 200;
            break;
        case 304:
            ce.Value = 450;
            break;
        case 310:
            ce.Value = 50;
            break;
        case 335:
            ce.Value = 1;
            break;
        default:
            break;
    }
});
bool result =
cuDriveFunctionInterface.HardwareProjection.ProjectMotorConfiguration(config, 0);
```

Die folgenden Beispiele zeigen, wie Sie die Motorkonfiguration für S120 und G220-Antriebe durchführen können:

Motorkonfiguration für S120 und G220-Antriebe projektieren

```

private bool S120ProjectMotorConfigurationTestWithName_ServoInductionMotor()
{
    m_Device.PlugNew(@"OrderNumber:6SL3120-1TE15-0Axx//10002", "", 65535); //momo
    DeviceItem mmDeviceItem = m_Device.DeviceItems[2];
    mmDeviceItem.PlugNew(@"OrderNumber:XMxxxxx-xxxxx-xxxx//Motor data input-Induction
motors", "", 65535);
    DriveObject mmDriveObject =
mmDeviceItem.GetService<DriveObjectContainer>().DriveObjects[0];
    DriveFunctionInterface mmDriveFunctionInterface =
mmDriveObject.GetService<DriveFunctionInterface>();
    HardwareProjection hwProjection = mmDriveFunctionInterface.HardwareProjection;
    bool result = false;
    try
    {
        MotorConfiguration motConfig = hwProjection.GetCurrentMotorConfiguration(0);
        if (motConfig.RequiredConfigurationEntries.Count == 0 ||
motConfig.OptionalConfigurationEntries.Count == 0)
        {
            return false;
        }
        motConfig.RequiredConfigurationEntries.ToList().ForEach(ce =>
        {
            //get all other properties of motor
            switch (ce.Name)
            {
                case "p305":
                    ce.Value = 1000;
                    break;
                case "p307":
                    ce.Value = 500;
                    break;
                case "p311":
                    ce.Value = 30;
                    break;
                case "p304":
                    ce.Value = 10;
                    break;
                case "p310":
                    ce.Value = 20;
                    break;
                case "p322":
                    ce.Value = 22;
                    break;
                case "p604":

```


Motorkonfiguration für S120 und G220-Antriebe projektieren

```
ce.Value = 30;
break;
case "p605":
ce.Value = 45;
break;
case "p350":
ce.Value = 64;
break;
case "p354":
ce.Value = 65;
break;
case "p356":
ce.Value = 66;
break;
case "p358":
ce.Value = 67;
break;
case "p360":
ce.Value = 68;
break;
default:
break;
}
});
motConfig.OptionalConfigurationEntries.ToList().ForEach(ce =>
{
switch (ce.Name)
{
case "p308":
ce.Value = 1;
break;
case "p338":
ce.Value = 6000;
break;
case "p348":
ce.Value = 20000;
break;
default:
break;
}
});
result = hwProjection.ProjectMotorConfiguration(motConfig, 0);
}
catch (Exception ex)
```

Motorkonfiguration für S120 und G220-Antriebe projektieren

```
{  
    Console.WriteLine(ex.Message + "\n" + ex.StackTrace);  
    TestInterface.Log.Error(ex.Message + "\n" + ex.StackTrace);  
}  
return result;  
}
```

5.23.5.16 Parameter lesen und schreiben

Das folgende Beispiel zeigt, wie Sie Werte von Antriebsparametern lesen und schreiben. Für den Zugriff benötigen Sie ein Antriebsobjekt.

Zugriff auf Parameter

```
using Siemens.Engineering.MC.Drives;
```

Zugriff auf Parameter

```
//Access a parameter via its name
DriveParameter parameter = driveObject.Parameters.Find("p5391[0]");

//Example of reading parameter attributes
if (parameter != null)
{
    Console.WriteLine("The Name of the parameter is : " + parameter.Name);
    Console.WriteLine("The value of the parameter is : " +
parameter.Value.ToString());
    Console.WriteLine("The minvalue of the parameter is : " +
parameter.MinValue);
    Console.WriteLine("The MaxValue of the parameter is : " +
parameter.MaxValue);
    Console.WriteLine("The Unit of the parameter is : " + parameter.Unit);

    //Example for write:
    parameter.Value = 60;
}

// Access a parameter via Number and ArrayIndex// Note that
// - arrayless parameters (e.g. p96 on G120) are indexed by -1
// - parameters that consist of only a bit array do not count as
//   array parameters, they are indexed by -1 as well and the
//   further bit values can be accessed by the 'Bits' property
//   of the given parameter (e.g. r2139 on G120). For further
//   information about accessing bit parameters see the next
//   section of this code snippet

DriveParameter r947_6 = driveObject.Parameters.Find(947, 6); // returns
r947[6]if (r947_6 != null)
{
    Console.WriteLine("The Name of the parameter is : " + r947_6.Name);
    Console.WriteLine("The value of the parameter is : " +
r947_6.Value.ToString());
}

//Accessing bit values
DriveParameter p2720 = driveObject.Parameters.Find(2720, 0);if (p2720 !=
null)
{
    // Note that in general, pXXX.Bits[YYY] is not necessarily equivalent to
    pXXX.YYY.
    // pXXX.Bits is an array of available bit values in ascending order by
    their names
    // and not an array of bit values indexed by their names.
    // For example: let the available bit values be [pXXX.0, pXXX.2, pXXX.3],
    then
    // pXXX.Bits[2] == pXXX.3, not pXXX.2

    DriveParameter p2720Bit1 = p2720.Bits[1]; // returns p2720[0].1
    if (p2720Bit1 != null)
    {
        Console.WriteLine("The name of the parameter is : " + p2720Bit1.Name);
        Console.WriteLine("The value of the second bit of the parameter is : "
+ p2720Bit1.Value.ToString());
    }
}
```

Zugriff auf Parameter

```

}

//Get the enum values of a parameter
DriveParameter r47 = driveObject.Parameters.Find("r47");
foreach (var enumItem in r47.EnumValueList)
{
    Console.WriteLine("Enum value: " + enumItem.Key.ToString() + " = " +
enumItem.Value);
}

```

Siehe auch

Contacts at Siemens (https://www.automation.siemens.com/aspa_app?ci=yes&lang=de)

5.23.5.17 Parameter online lesen und schreiben

Das folgende Beispiel zeigt, wie Sie eine Liste mit den verfügbaren Online-Parametern erhalten. Für den Zugriff benötigen Sie ein Online-Antriebsobjekt.

Der Lese- und Schreibzugriff auf einzelne Online-Parameter aus der Parameterliste ist identisch wie im Beispiel Parameter lesen und schreiben (Seite 1182).

Zugriff auf Online-Parameter

```

using Siemens.Engineering.MC.Drives;
DeviceItem item = ... //device item of the CU (e.g. :
project.Devices[0].Items[0].Items[0])
OnlineDriveObject onlineDriveObject =
item.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];
if (onlineDriveObject != null)
{
    var parameters = onlineDriveObject.Parameters;
}

```

5.23.5.18 Parametrierung speichern

Das folgende Beispiel zeigt, wie Sie die Parametrierung aus S120, S210 oder G120-Antrieben ermitteln können.

Parametrierung online aus den Antrieben ermitteln

```

using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;

```

Parametrierung online aus den Antrieben ermitteln

```

OnlineDriveObject onlineDriveObject =
item.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];
OnlineDriveFunctionInterface onlineDfi =
onlineDriveObject.GetService<OnlineDriveFunctionInterface>();
DriveDomainFunctions driveDomainFunctions = onlineDfi.DriveDomainFunctions;

// onlineDfi can be null in case of the actual onlineDriveObject
// does not support it or if the device is offline.

// driveDomainFunctions can be null, if the actual onlineDriveObject
// does not support it.

// For example: On G120 and S210 drives, you have to use the CU to get the
onlineDriveObject.
// Please, pay attention at S120 drives. Here you can use any module (CU, MotorModul,
lineModul) as onlineDriveObject but preferable the CU, because it will run on CU!

```

Das folgende Beispiel zeigt, wie Sie die Parametrierung bei S120, S210 oder G120-Antrieben von RAM nach ROM kopieren und damit netzausfallsicher speichern können.

Sicherung RAM nach ROM durchführen

```

using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;

//In case of G120 device.
DeviceItem cuDeviceItem = m_Device.DeviceItems[1];
OnlineDriveObject cuDriveObject =
cuDeviceItem.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];

//In case of S120 and S210 device you should generally get the CU driveobject.
DeviceItem cuDeviceItem = m_Device.DeviceItems[0];OnlineDriveObject cuDriveObject =
cuDeviceItem.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];

// To use the function, you have to use the OnlineDriveFunctionInterface.
OnlineDriveFunctionInterface cuDriveFunctionInterface =
cuDriveObject.GetService<OnlineDriveFunctionInterface>();
DriveDomainFunctions driveDomainFunctions = cuDriveFunctionInterface.DriveDomainFunctions;

// This function will perform a RAM to ROM copy on all device.
// It is important that the CU should be in online state before you call this function
anyway the program will give an exception.
// This call may take a few moments before be finished.
bool result = DriveDomainFunctions.PerformRAMtoROMCopyAllDriveObject();

```

5.23.5.19 SecureString-Passwort vergeben

Mit der Methode SetPassword() können Sie Passwörter zur sicheren Kommunikation beim Download vergeben. Der Parameter muss ein SecureString sein.

Das folgende Beispiel zeigt, wie Sie ein SecureString-Passwort vergeben.

SecureString-Passwort vergeben

SecureString-Passwort vergeben

```

public void SetPassword(DownloadPasswordConfiguration
downloadPasswordConfiguration)
{
    if (downloadPasswordConfiguration != null)
    {
        SecureString s = GetSecureString();
        downloadPasswordConfiguration.SetPassword(s);
        return;
    }
}

private SecureString GetSecureString()
{
    SecureString securePwd = new SecureString();
    ConsoleKeyInfo key;

    Console.Write("Enter password: ");
    do
    {
        key = Console.ReadKey(true);

        // Ignore any key out of range.
        if (((int)key.Key) >= 65 && ((int)key.Key) <= 90)
        {
            // Append the character to the password.
            securePwd.AppendChar(key.KeyChar);
            Console.Write("*");
        }
        // Exit if Enter key is pressed.
    } while (key.Key != ConsoleKey.Enter);

    return securePwd;
}

```

Siehe auch

Verschlüsselte Kommunikation mit SecureString-Passwörtern (Seite 1129)

5.23.5.20 Safety Integrated (Prüfsummen) konfigurieren (ab SINAMICS FW V6.1)

Die folgenden Beispiele zeigen, wie Sie für SINAMICS-Antriebe ab Firmware-Version V6.1 mit Openness die Safety Integrated Functions über Prüfsummen konfigurieren. Diese Funktion ist nur Offline-Modus verfügbar.

Vor dem Zugriff auf `UpdateChecksums` muss über `DriveObject` das Antriebsojekt ermittelt werden (Seite 1155).

Safety Integrated über Prüfsummen konfigurieren (nur offline)

```

using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;

```

Safety Integrated über Prüfsummen konfigurieren (nur offline)

```
DriveObject cuDriveObject =  
cuDeviceItem.GetService<DriveObjectContainer>().DriveObjects[0];  
DriveFunctionInterface driveFunctionInterface =  
cuDriveObject .GetService<DriveFunctionInterface>();  
  
if (driveFunctionInterface.SafetyCommissioning == null)  
{  
    return false;  
}  
  
//UpdateCheckSums throws exception in case, if the specified CoreObject is null or not  
offline  
bool retValue = driveFunctionInterface.SafetyCommissioning.UpdateCheckSums();  
  
// onlineDfi can be null in case of the actual onlineDriveObject  
// does not support it or if the device is online.  
  
// G120 and S120 drives(drives with SINAMICS Firmware version older than V6.1) do not  
support it.  
// For G120 drives(CU250S-2_PN), driveFunctionInterface.SafetyCommissioning returns  
'UpdateCheckSums is not supported for 2nd gen drives'  
// For S120 acx drives, driveFunctionInterface.SafetyCommissioning returns null
```

5.23.5.21 Safety Integrated-Telegramme verwenden

Das folgende Beispiel zeigt, wie Sie Safety Integrated-Telegramme (z. B. 30) in Openness verwenden können.

Safety Integrated-Telegramme verwenden

```
using Siemens.Engineering.MC.Drives;  
TelegramComposition telegrams = drvObj.Telegrams;  
  
//Add safety telegram  
const int tgrmNumber = 30;  
drvObj.Telegrams.InsertSafetyTelegram(tgrmNumber);  
  
//Find safety telegram  
Telegram safetyTgrm = drvObj.Telegrams.Find(TelegramType.SafetyTelegram);  
  
// Get and set safety telegram attributes  
uint watchdogTime = (uint)safetyTgrm.GetAttribute("Failsafe_FMonitoringtime");  
  
safetyTgrm.SetAttribute("Failsafe_FMonitoringtime", 300);  
  
const int newSafetyTelegramNumber= 900;  
if (safetyTgrm.CanChangeTelegram(newSafetyTelegramNumber))  
{  
    safetyTgrm.TelegramNumber = newSafetyTelegramNumber;  
}  
  
//Remove Safety telegram  
drvObj.Telegrams.EraseTelegram(TelegramType.SafetyTelegram);
```

5.23.5.22 SIMOGEAR-Artikelnummer für G115D-Antriebe setzen

Das folgende Beispiel zeigt, wie Sie für G115D-Antriebe die SIMOGEAR-Artikelnummer einstellen können.

SIMOGEAR-Artikelnummer für G115D-Antriebe einstellen

```
using Siemens.Engineering.MC.Drives;
Dvar commissioning= dfi.Commissioning;
var commissioning= dfi.Commissioning;
//Commissioning can be null if the actual driveobject does not support the Commissioning
(e.g. SINAMICS S devices).

//Setting the article number of the SimoGear
commissioning.SetSimoGearMlfb("2KJ8001-2EG20-4DG1-D0X");
```

Siehe auch

Commissioning (Seite 1133)

5.23.5.23 Technology Extensions verwenden

Die folgenden Beispiele zeigen, wie Sie Technology Extensions verwenden können.

TechnologyExtensionPackage installieren

```
string packageFilePath = @"Y:\TRCDATA_V1_1_0_1.tec";

TechnologyExtensionInstallationProvider tecInstallationProvider =
TiaPortal.GetService<TechnologyExtensionInstallationProvider>();
FileInfo packageFile = new FileInfo(packageFilePath);

try
{
tecInstallationProvider.Install(packageFile);
}
catch (EngineeringException ex)
{
Console.WriteLine(ex.Message + "\n" + ex.StackTrace);
}
```

Technology Extension aktivieren

```
TechnologyExtension tecExtension= ...
tecExtension.Activate();
return tecExtension.IsActivated;
```

Technology Extension deaktivieren

```
TechnologyExtension tecExtension=...
tecExtension.Deactivate();
return tecExtension.IsActivated;
```


TechnologyExtensionPackage deinstallieren

```
string tecPackageIdentifier = "TRCDATA/1100802";
bool confirmUninstallInUse = true;
TechnologyExtensionInstallationProvider tecInstallationProvider =
TiaPortal.GetService<TechnologyExtensionInstallationProvider>();
TechnologyExtensionPackage tecPackage =
tecInstallationProvider.TechnologyExtensionPackages.Find(tecPackageIdentifier);

if (tecPackage == null)
{
    Console.WriteLine("the specified TechnologyExtensionPackage '{tecPackageIdentifier}' was
not found");
    return;
}

try
{
    tecInstallationProvider.Uninstall(tecPackage.Identifier, confirmUninstallInUse);
}
catch (EngineeringException ex)
{
    Console.WriteLine(ex.Message + "\n" + ex.StackTrace);
}
```

DriveObjectContainer eines SINAMICS-Geräts auslesen

```
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.MC.Drives

Device device = ...;
foreach (DeviceItem deviceItem in device.DeviceItems)
{
    DriveObjectContainer doContainer = deviceItem.GetService<DriveObjectContainer>();
    // do something
}
```

Zu einem DriveObject navigieren

```
Device device = ...;
foreach (DeviceItem deviceItem in device.DeviceItems)
{
    DriveObjectContainer doContainer = deviceItem.GetService<DriveObjectContainer>();
    DriveObjectComposition driveObjects = doContainer.DriveObjects;
    foreach (DriveObject driveObject in driveObjects)
    {
        // do something with the DriveObject
    }
}
```

Zu einer Technology Extension navigieren

```
DriveObject driveObject = ...
TechnologyExtensionContainer tecContainer =
driveObject.GetService<TechnologyExtensionContainer>();
TechnologyExtensionComposition tecExtensions = tecContainer.TechnologyExtensions;
foreach (TechnologyExtension tecExtension in tecExtensions)
{
// do something with the TechnologyExtensions
}
```

TechnologyExtension anhand von TechnologyExtension.Name finden

```
DriveObject driveObject = ...;
TechnologyExtensionContainer tecContainer =
driveObject.GetService<TechnologyExtensionContainer>();
TechnologyExtension tecExtension = tecContainer.TechnologyExtensions.FirstOrDefault(x =>
x.Name == tecExtensionName);
// do something with the TechnologyExtension
```

Siehe auch

[TechnologyExtension \(Seite 1145\)](#)

5.23.5.24 Technologieobjekte über Telegrammobjekte mit Hardware-Modulen verbinden

Mit der Methode Connect können Sie Technologieobjekte über Telegrammobjekte mit Hardware-Modulen verbinden.

Technologieobjekte über Telegrammobjekte mit Hardware-Modulen verbinden

```
private bool Test()
{
    Device plcStation = m_Project.Devices[0];
    DriveObject driveObject =
m_Project.Devices.ElementAt(1).DeviceItems[2].GetService<DriveObjectContai
ner>().DriveObjects[0];
    Telegram telegram = driveObject.Telegrams[0];

    DeviceItem plcDevice = plcStation.DeviceItems[1];
    SoftwareContainer softwareContainer =
plcDevice.GetService<SoftwareContainer>();
    Software software = softwareContainer.Software;

    TechnologicalInstanceDB techObj0 =
((Siemens.Engineering.SW.PlcSoftware)software).TechnologicalObjectGroup.Te
chnologicalObjects[0];
    AxisHardwareConnectionProvider axisHardwareConnectionProvider0 =
techObj0.GetService<AxisHardwareConnectionProvider>();
    AxisEncoderHardwareConnectionInterface
axisEncoderHardwareConnectionInterfaces0 =
axisHardwareConnectionProvider0.ActorInterface;

    try
    {
        axisEncoderHardwareConnectionInterfaces0.Connect(telegram); //
Connect API
    }
    catch (Exception e)
    {
    }
}
```

Technologieobjekte mit einem Hardware-Modul über ein Telegrammobjekt und ConnectOption verbinden

```

private bool Test()
{
    Device plcStation = m_Project.Devices[0];
    DriveObject driveObject =
m_Project.Devices.ElementAt(1).DeviceItems[2].GetService<DriveObjectContai
ner>().DriveObjects[0];
    Telegram telegram = driveObject.Telegrams[0];

    DeviceItem plcDevice = plcStation.DeviceItems[1];
    SoftwareContainer softwareContainer =
plcDevice.GetService<SoftwareContainer>();
    Software software = softwareContainer.Software;

    TechnologicalInstanceDB techObj0 =
((Siemens.Engineering.SW.PlcSoftware)software).TechnologicalObjectGroup.Te
chnologicalObjects[0];
    AxisHardwareConnectionProvider axisHardwareConnectionProvider0 =
techObj0.GetService<AxisHardwareConnectionProvider>();
    AxisEncoderHardwareConnectionInterface
axisEncoderHardwareConnectionInterfaces0 =
axisHardwareConnectionProvider0.ActorInterface;

    try
    {
        //Connect API
        axisEncoderHardwareConnectionInterfaces0.
Connect(telegram,
Siemens.Engineering.MC.Drives.Enums.ConnectOption.Default); //Default = 0,
AllowAllModules = 1
        //Or the following way can also be used
        axisEncoderHardwareConnectionInterfaces0.Connect(telegram, 0); //
Default = 0, AllowAllModules = 1
    }
    catch (Exception e)
    {
    }
}

```

Siehe auch

HardwareProjection (Seite 1139)

5.23.5.25 Telegramme einfügen und erweitern

Das folgende Beispiel zeigt, wie Sie eine Erweiterung einfügen und die Größe eines Standardtelegramms verändern. Für den Zugriff benötigen Sie ein Antriebsobjekt.

Erweiterung einfügen und die Größe eines Standardtelegramms ändern

```
using Siemens.Engineering.MC.Drives;
```

Erweiterung einfügen und die Größe eines Standardtelegramms ändern

```

TelegramComposition telegrams = drvObj.Telegrams;
Telegram telegram = telegrams.Find(TelegramType.MainTelegram);

Console.WriteLine("The Cu has the telegram: " + telegram.TelegramNumber);
Console.WriteLine("The Setpoint channel-specific size of the telegram is: "
+ telegram.GetOutputSize());

foreach (var address in telegram.Addresses)
{
    if (address.IoType == AddressIoType.Output)
    {
        Console.WriteLine("The Setpoint channel-specific starting address of
the connected PLC is: " + address.StartAddress);
    }
    else if (address.IoType == AddressIoType.Input)
    {
        Console.WriteLine("The Actual value channel-specific starting address
of the connected PLC is: " + address.StartAddress);
    }
}

// Add an additional Telegram
if (drvObj.Telegrams.CanInsertAdditionalTelegram(2,4))
{
    drvObj.Telegrams.InsertAdditionalTelegram(2,4);
}

// Add a 3 word extension to the main telegram
Telegram mainTelegram == drvObj.Telegrams.Find(TelegramType.MainTelegram);
Int32 newSize = mainTelegram.GetSize(AddressIoType.Input) + 3;
if (mainTelegram.CanChangeSize(AddressIoType.Input, newSize, true))
{
    mainTelegram.ChangeSize(AddressIoType.Input, newSize, true)
}

```

5.23.5.26 Torque-Telegramme verwenden

Das folgende Beispiel zeigt, wie Sie Torque-Telegramme (z. B. 750) in Openness verwenden können.

Torque-Telegramme verwenden

```

using Siemens.Engineering.MC.Drives;
const int torqueTelegramNumber = 750;
TelegramComposition telegrams = drvObj.Telegrams;

// Add a new torque telegram
if (drvObj.Telegrams.CanInsertTorqueTelegram(torqueTelegramNumber))
{
    drvObj.Telegrams.InsertTorqueTelegram(torqueTelegramNumber);
}

// Find torque telegram
Telegram torqueTelegram = drvObj.Telegrams.Find(TelegramType.TorqueTelegram);

```

5.23.5.27 UMAC für Antriebe aktivieren/deaktivieren

Die folgenden Beispiele zeigen, wie Sie die Funktion Benutzerverwaltung und Zugriffskontrolle (UMAC, User Management and Access Control) für SINAMICS-Antriebe ab Firmware-Version V6.1 mit Openness aktivieren und deaktivieren. UMAC kann mit Openness nur im Offline-Modus aktiviert oder deaktiviert werden.

Vor dem Zugriff auf `UmacConfiguration` muss über `DriveObject` das Antriebsojekt ermittelt werden (Seite 1155).

UMAC aktivieren (nur offline)

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.Security;
DeviceItem cuDeviceItem = device.DeviceItems[0];
DriveObject cuDriveObject =
cuDeviceItem.GetService<DriveObjectContainer>().DriveObjects[0];
Security cusecurity=cuDriveObject.Security;
if(cuSecurity == null)
//For G120 drives (CU250S-2_PN), as UmacConfiguration is not supported for drives with
SINAMICS Firmware version older than V6.1, the value of 'Security' object will be null
{
    return false;
}
UmacConfiguration umacConfiguration = cusecurity.UmacConfiguration;

//Activate the UMAC for drive
bbool result = umacConfiguration.Activate();
```

UMAC deaktivieren (nur offline)

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.Security;
//For online
OnlineDriveObject cuOnlineDriveObject =
device.DeviceItems[itemPosition].GetService<OnlineDriveObjectContainer>().OnlineDriveObje
cts[0];

//Activate the DriveDataEncryption for the drive
bool result = cuOnlineDriveObject.Security.DriveDataEncryption.Activate(curpassword);

//Deactivate the DriveDataEncryption for drive
bool result = cuOnlineDriveObject.Security.DriveDataEncryption.Deactivate(oldpassword);

//Deactivate the UMAC for the drive

bool result = umacConfiguration.Deactivate();
```

5.23.5.28 Verschlüsselung der Antriebsdaten/DDE aktivieren (ab SINAMICS FW V6.1)

Die folgenden Beispiele zeigen, wie Sie die Verschlüsselung der Antriebsdaten (DDE, Drive Data Encryption) für SINAMICS-Antriebe ab Firmware-Version V6.1 mit Openness aktivieren und deaktivieren.

Vor dem Zugriff auf `DriveDataEncryption` muss über `DriveObject` das Antriebsojekt ermittelt werden (Seite 1155).

Verschlüsselung der Antriebsdaten / DDE offline aktivieren und deaktivieren

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.Security;

//For offline
DriveObject driveObject =
device.DeviceItems[itemPosition].GetService<DriveObjectContainer>().DriveO
bjects[0];

//Activate the DriveDataEncryption for the drive
bool result =
driveObject.Security.DriveDataEncryption.Activate(curpassword);

//Deactivate the DriveDataEncryption for drive
bool result =
driveObject.Security.DriveDataEncryption.Deactivate(oldpassword);
```

Verschlüsselung der Antriebsdaten / DDE online aktivieren und deaktivieren

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.Security;

//For online
OnlineDriveObject cuOnlineDriveObject =
device.DeviceItems[itemPosition].GetService<OnlineDriveObjectContainer>().
OnlineDriveObjects[0];

//Activate the DriveDataEncryption for the drive
bool result =
cuOnlineDriveObject.Security.DriveDataEncryption.Activate(curpassword);

//Deactivate the DriveDataEncryption for drive
bool result =
cuOnlineDriveObject.Security.DriveDataEncryption.Deactivate(oldpassword);
```

5.23.5.29 Werkseinstellungen wiederherstellen

Das folgende Beispiel zeigt, wie Sie für S120, S210 oder G120-Antriebe die Werkseinstellung wiederherstellen können.

Werkseinstellungen wiederherstellen

```
using Siemens.Engineering.MC.Drives;
using Siemens.Engineering.MC.Drives.DFI;
using Siemens.Engineering.MC.Drives.Enums;
```

5.24 Funktionen für SIMATIC Drive Controller

Werkseinstellungen wiederherstellen

```
OnlineDriveObject onlineDriveObject = ...
OnlineDriveFunctionInterface onlineDfi =
onlineDriveObject.GetService<OnlineDriveFunctionInterface>();
DriveDomainFunctions driveDomainFunctions = onlineDfi.DriveDomainFunctions;

// onlineDfi can be null in case of the actual onlineDriveObject
// does not support it or if the device is offline.
// saveParametrization can be null, if the actual onlineDriveObject
// does not support it.
// For example: On G120 and S210 drives, you have to use the CU to get the
onlineDriveObject.
// S120 drives, you can use any module (CU, MotorModul, lineModul) as onlineDriveObject but
preferable the CU.

//In case of G120 device
DeviceItem cuDeviceItem = g120Device.DeviceItems[1];
OnlineDriveObject cuDriveObject =
cuDeviceItem.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];

//In case of S120 and S210 devices you should generally get the CU driveobject
DeviceItem cuDeviceItem = s120Device.DeviceItems[0]
OnlineDriveObject cuDriveObject =
cuDeviceItem.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];

// To use the function, you have to use the OnlineDriveFunctionInterface
OnlineDriveFunctionInterface cuDriveFunctionInterface =
cuDriveObject.GetService<OnlineDriveFunctionInterface>();
DriveDomainFunctions driveDomainFunctions = cuDriveFunctionInterface.DriveDomainFunctions;

// This function will perform a FactoryReset for S120 drives
// It is important that the CU should be in online state before you call this function
// anyway the program will give an exception.
// This call may take a few moments before be finished.
bool result = driveDomainFunctions.PerformFactoryReset(ResetMode.SafetyParameterReset);

//G120 drives have 2 flags for the ResetMode
The flag ParameterReset is used to reset the norml parameters.
The flag SafetyParameterReset is used to reset the safety parameters
```

5.24 Funktionen für SIMATIC Drive Controller

5.24.1 Zugreifen auf PLC und Kanaleigenschaften von X142

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness zugreifen auf:

- Parameter der PLC
- Parameter der X142 Onboard-I/Os für jeden Kanal

PLC-Parameter

Zusätzlich zu den SIMATIC S7-1500 PLC-Attributen bietet der SIMATIC Drive Controller die folgenden für das Gerät spezifischen Attribute:

Name der Eigenschaft	Datentyp	Schreibbar	Zugriff
TimeSynchronizationDriveActive	Bool	r/w	Dynamisches Attribut

TimeSynchronizationDriveActive ist eine Enum mit dem folgenden Wert:

Eigenschaft	Beschreibung	Wert
TimeSynchronizationDriveActive	Deaktiviert	0
	Aktiviert	1

X142-Parameter

SIMATIC Drive Controller bietet die folgenden für die Antriebsregelung spezifischen Attribute für die X142 Onboard-I/Os.

X142-Basisparameter

Attribut	Datentyp	Schreibbar	Beschreibung
PulseWidthModulationFrequency	UInt64	r/w	Basisfrequenz der Impulsdauermodulation

PulseWidthModulationFrequency ist eine Enum-Eigenschaft mit dem folgenden Wert:

Enum	Wert
goverance	0
Value01Kilohertz	1
Value02Kilohertz	2
Value04Kilohertz	3
Value08Kilohertz	4
Value16Kilohertz	5

X142-Kanalparameter

Attribut	Datentyp	Schreibbar	Beschreibung
IoType	Siemens.Engineering.HW.Channello-Type	r	Wert: Complex Komplexe I/O-Typen, z. B. für technologische Kanäle
Type	Siemens.Engineering.HW.Channel-Type	r	Wert: Technology Der Kanaltyp ist Technologie
Number	Int32	r	Die Kanalnummer ist 0 ... 7
OperatingMode	UInt64	r/w	Auswahl der Betriebsarten für die einzelnen Kanäle
Invert	Bool	r/w	Das Signal kann umgekehrt werden
InputDelay	UInt64	r/w	Um Störungen zu unterdrücken, kann eine Eingangsverzögerung von 1 µs oder 125 µs für den Eingangsfilter der Digitaleingabe eingestellt werden
HighSpeedOutput	Bool	r/w	Ist die Option für die Hochgeschwindigkeitsausgabe eingestellt, wird der Digitalausgang abwechselnd zwischen 24 V DC und Erde geschaltet. Dies ermöglicht extrem steile Flanken (Ausgangsverzögerung von 1 µs)

OperatingMode ist eine Enum-Eigenschaft mit dem folgenden Wert:

Enum	Wert
DigitalInput	52
DigitalOutput	276
TimerDQ	8
TimerDI	34
OversamplingInput	283
OversamplingOutput	284
PeriodMeasurement	31
PulseWidthModulation	7

InputDelay ist eine Enum-Eigenschaft mit dem folgenden Wert:

Enum	Wert
Value001Microseconds	33
Value125Microseconds	34

Programmcode

Um die Werte von dynamischen Attributen abzurufen und einzustellen, ändern Sie folgenden Programmcode:

```
// To create a SIMATIC Drive Controller
var driveController =
MyTiaPortal.Projects.First().Devices.CreateWithItem("OrderNumber:6ES7 615-4DF10-0AB0/
V2.8","PLC_1", "PLC_1");
// To get channels of DI/DQ 8x24VDC_1
DeviceItem di_dq = driveController.DeviceItems.First(x => x.Classification ==
Classification.CPU).DeviceItems[7];
ChannelComposition channels = di_dq.Channels;
foreach (Channel channel in channels)
{
... // work with the channel
}
// To get properties of a channel:
Channel channel = channels[0];
int channelNumber = channel.Number;
// To get values of dynamic attributes:
var isInvert = channel.GetAttribute("Invert");
var operatingMode = channel.GetAttribute("OperatingMode");

// To set values of a channel:
channel.SetAttribute("OperatingMode",OperatingMode.TimerDQ);
```

5.24.2 Zugreifen auf die integrierten Eigenschaften von PROFIdrive

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness ein Subnetz für PROFIdrive Integrated erstellen und auf die Eigenschaften eines PROFIdrive Integrated zugreifen.

Eigenschaft

Auf die folgenden Eigenschaften von PROFIdrive Integrated mit isochron kann über TIA Portal Openness zugegriffen werden:

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w	dynamisch	Name des Subnetzes
NetType	NetType	r	modelliert	Typ des Subnetzes
IsochronousMode	Bool	r	dynamisch	Konstante Buszykluszeit aktiviert
SourceCycleTime*	Int32	r/w	dynamisch	Die Zykluszeit der Busquelle
DpCycleTime	double	r/w	dynamisch	Die DP-Zykluszeit
IsochronousTiToAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der Werte Ti (Lesen von Prozesswerten) und To (Ausgeben von Prozesswerten) aktiviert sind.
IsochronousTi	double	r/w	dynamisch	Time Ti (Zeit zum Lesen von Prozesswerten)
IsochronousTo	double	r/w	dynamisch	Time To (Zeit zum Ausgeben von Prozesswerten)

*SourceCycleTime unterstützt die folgenden Enum-Werte:

Enum-Name	Wert
Manual	0
AutomaticMinimum	1
LocalSendClock	2
ProfinetSendClock	3

Programmcode

Um auf die Eigenschaften von PROFIdrive Integrated zuzugreifen, ändern Sie folgenden Programmcode:

```
Siemens.Engineering.HW.SubnetComposition subnetComposition =
MyTiaPortal.Projects.First().Subnets;
Subnet profiDriveSubnet = subnetComposition.Create("System:Subnet.ProfidriveIntegrated",
"PROFIdrive Integrated_1");
//Couple PROFIdrive Integrated to X150 (PROFINET)
var profinetX150 = driveController.DeviceItems.First(x => x.Classification ==
Classification.CPU).DeviceItems[4].GetService<NetworkInterface>().Nodes.First();
profinetX150.ConnectToSubnet(profiDriveSubnet);
```

5.24.3 Erstellen von SINAMICS Integrated

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness ein SINAMICS Integrated erstellen:

Programmcode

```
Siemens.Engineering.HW.DeviceComposition deviceComposition = project.Devices;  
Device S120 = deviceComposition.CreateWithItem(@"OrderNumber:6ES7615-xDS1x-0xxx/V5.2/  
S120", "Integrated_1", "SINAMICS S_1");
```

5.24.4 Abrufen der Adresse des zentralen Bausteins (X142)

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um die Adresse des zentralen Bausteins X142 abzurufen, ändern Sie folgenden Programmcode:

```
//Simatic Drive Controller is usually created in a DeviceGroup (group)  
//Getting headmodule of the SIMATIC Drive Controller  
DeviceItem plcHeadModule = group.Devices[0].DeviceItems[1];  
//Getting address of central module (X142)  
Address x142address = plcHeadModule.DeviceItems.First(x => x.PositionNumber ==  
8).Addresses[0];  
//Turn on isochronous mode  
x142address.SetAttribute("IsochronousMode", true);
```

5.24.5 Abrufen der Adresse des Antriebsregelungstelegramms

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Programmcode

Um die Adresse des Antriebsregelungstelegramms abzurufen, ändern Sie folgenden Programmcode:

```
//Simatic Drive Controller is usually created in a DeviceGroup (group)
//Getting headmodule of the S120
DeviceItem s120HeadModule = group.Devices[1].DeviceItems[0];
//Getting address of drive controller telegram
DriveObjectContainer driveObjectService =
S120DeviceItem.GetService<DriveObjectContainer>();
Telegram telegram = driveObjectService.DriveObjects.First().Telegrams.First();
Address telegramAddress = telegram.Addresses[0];
//Turn on isochronous mode
telegramAddress.SetAttribute("IsochronousMode", true);
```

5.25 Funktionen für DCC

5.25.1 Einführung

Über TIA Portal-Openness automatisieren Sie das Engineering und steuern das TIA Portal über ein von Ihnen selbst erstelltes Programm.

Für dieses selbst erstellte Programm finden Sie in dieser Hilfe viele Informationen und Beispielcodes. Auch für die TIA Portal-Anwendung "DCC" können Sie eigene Programme zusammenstellen und anwenden.

Bevor Sie für DCC aus den nachfolgend gelisteten Beispielcodes ein eigenes Programm zusammenstellen, beachten Sie bitte die übergreifenden Informationen zu Openness, die Sie unter folgenden Stichworten in dieser Hilfe finden:

- Voraussetzungen für TIA Portal Openness
- TIA Portal Openness installieren
- Auf das TIA Portal zugreifen
- TIA Portal Openness-Objektmodell
- Programmierschritte

5.25.2 DCC Openness

Einleitung

Mit TIA Portal Openness V18 programmieren Sie Anwendungen, die das Engineering im TIA Portal automatisieren.

Funktionen von Openness in Verbindung mit SINAMICS DCC

Folgende Funktionen sind in der Version V18 implementiert:

- Pläne/Sub-Pläne/Blöcke erstellen/löschen
- Pläne/Blöcke umbenennen
- Kommentare bearbeiten (Plan, Block, Pin)
- Pin-Werte ändern
- Blöcke/Sub-Pläne anordnen
- Blöcke als Vorgänger setzen
- Ablaufreihenfolge von Plänen/Blöcken ändern
- Parameter inklusive Parametertext veröffentlichen/bearbeiten
- Pin-zu-Pin-Verbindungen erstellen/löschen
- Nach Blöcken/Sub-Plänen suchen
- DCB-Extension-Bibliotheken prüfen (ob importiert und wo verwendet)
- Pläne importieren (Seite 1223)
- Einen Plan exportieren (Seite 1223) oder alle Pläne (Ordner "Pläne") exportieren (Seite 1216)
- DCB Extension-Bibliotheken in die Projektbibliothek importieren (Seite 1216)
- Pläne anhand des Namens finden (Seite 1220)
- Pläne aus dem Ordner "Pläne" löschen (Seite 1222)
- Ablaufreihenfolge der Blöcke innerhalb eines Plans optimieren (Seite 1215)

Hinweis

Leistung optimieren

Für eine optimale Leistung bei der Nutzung von Openness wird empfohlen, betroffene Editoren (z. B. den DCC-Editor für Pläne, die Parameterliste usw.) zu schließen, solange eine aktive TIA Portal-Instanz mit Benutzeroberfläche geöffnet ist.

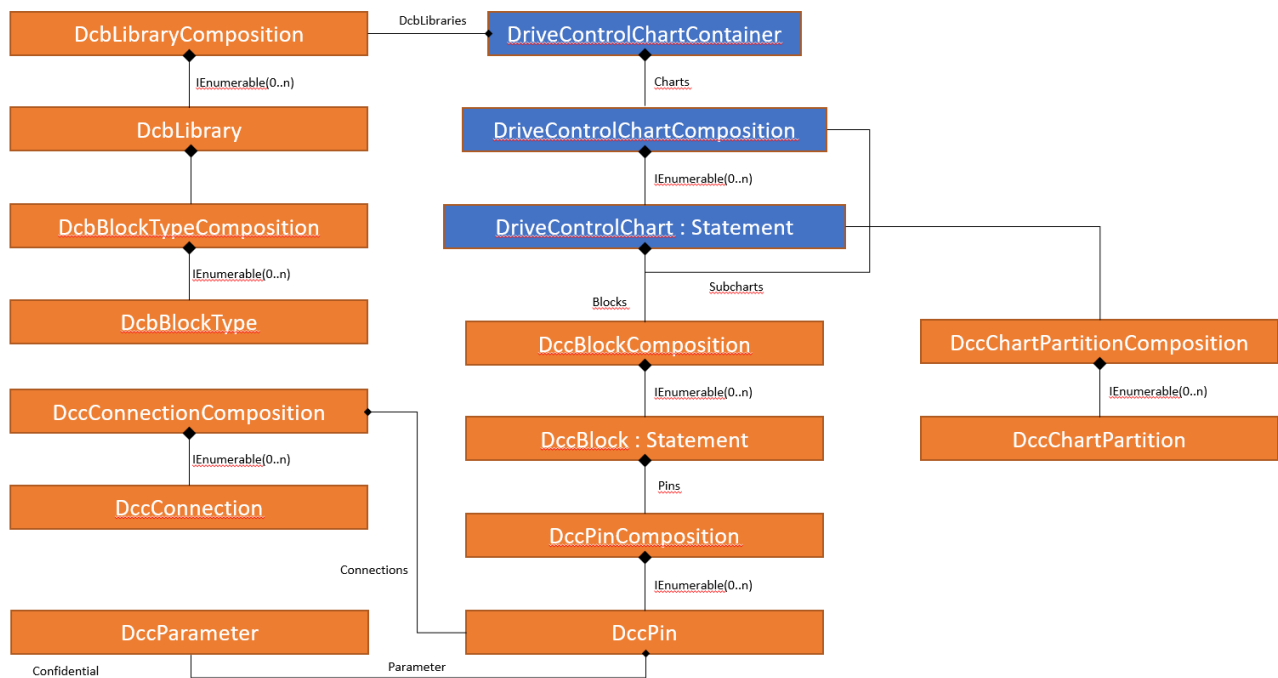
Weitere Informationen

Siehe TIA Informationssystem „Openness: Projekterstellung automatisieren“.

5.25.3 Objektmodell DCC Openness

Übersicht

Das folgende Diagramm beschreibt das DCC Openness-Objektmodell:



5.25.4 Referenzen

5.25.4.1 DccBlock

DccBlock Features

DccBlock entspricht einem DCC-Block innerhalb eines DCC-Plans.

Die folgende Tabelle beschreibt die **Attribute** von DccBlock:

Alle Attribute von Statement sowie die folgenden.

Attributname	Datentyp	Schreibbar	Beschreibung	Exceptions
GenericInputsNumber	ushort	Schreibbar	Die Nummer des generischen Eingangs-Pin des Blocks.	DccInvalidValueException

Die folgende Tabelle beschreibt die **Navigatoren** von DccBlock:

Name	Datentyp	Schreibbar	Beschreibung
Pins	DccPinComposition	Nur lesen	Gibt alle Pins eines Blocks zurück.

Die folgende Tabelle beschreibt die **Methoden** von DccBlock:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Delete	-	-	Löscht einen Block aus einem Plan.	DccException

5.25.4.2 DccBlockComposition

DccBlockComposition Features

DccBlockComposition entspricht einem Set von DCB-Blöcken innerhalb eines Plans oder Sub-Plans.

Die folgende Tabelle beschreibt die **Methoden** von DccBlockComposition:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Create	DccBlock	string name string Block-type	Erstellt einen neuen DCC-Block im Plan mit dem angegebenen Namen und Blocktyp-Namen.	DccInvalidNameException, DccTextTooLongException, DccNameAlreadyUsedException, DccBlockTypeNotFoundException
Create	DccBlock	string name string Block-type string LibraryName	Erstellt einen neuen DCC-Block im Plan mit dem angegebenen Namen und verwendet den Blocktyp-Namen aus der gewählten DCB-Bibliothek.	DccBlockCreationInvalidNameException, DccBlockCreationNameTooLongException, DccBlockCreationBlocktypeNotFoundException
Find	DccBlock	string name	Findet einen DCC-Block anhand des Namens.	-

5.25.4.3 DcbLibrary

DcbLibrary

DcbLibrary entspricht einer DCB-Bibliothek, die in einem DCC-Plan-Container verwendet wird.

Die folgende Tabelle beschreibt die **Attribute** von DcbLibrary:

Attributname	Datentyp	Schreibbar	Beschreibung
LibraryName	string	Nur lesen	Der Name der DCB-Bibliothek.
Version	system.Version	Nur lesen	Die Version der DCB-Bibliothek.

Die folgende Tabelle beschreibt die **Navigatoren** von DcbLibrary:

Name	Datentyp	Schreibbar	Beschreibung
DcbBlockTypes	DcbBlockTypeComposition	Nur lesen	Gibt alle Blocktypen zurück, die in der DCB-Bibliothek verwendet werden.

5.25.4.4 DcbBlockType

DcbBlockType

DcbBlockType entspricht einem DCB-Blocktyp, der in einer DCB-Bibliothek verfügbar ist.

Die folgende Tabelle beschreibt die **Attribute** von DcbBlockType:

Attributname	Datentyp	Schreibbar	Beschreibung
Name	string	Nur lesen	Der Name des DCB-Blocktyps.
Description	string	Nur lesen	Eine Kurzbeschreibung des DCB-Blocktyps.

5.25.4.5 DcbLibraryComposition

DcbLibraryComposition Features

DcbLibraryComposition entspricht einem Set von DCB-Bibliotheken innerhalb eines bestimmten des DCC Plan-Containers.

Die folgende Tabelle beschreibt die **Methoden** von DcbLibraryComposition:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Find	DcbLibraryComposition	string libraryName	Findet eine DcbLibrary anhand des Namens.	-

5.25.4.6 DccConnection

DccConnection Features

DccConnection entspricht einer internen oder externen Verbindung in einem DCC-Plan, bei der entweder die Senke, die Quelle oder beide ein DCC-Objekt sind (Pin oder Parameter).

Die folgende Tabelle beschreibt die **Navigatoren** von DccConnection:

Name	Datentyp	Schreibbar	Beschreibung
Sink	IEngineeringObject	Nur lesen	Die Senke einer Verbindung. Kann sein: DccPin.
Source	IEngineeringObject	Nur lesen	Die Quelle einer Verbindung. Kann sein: DccPin.

Die folgende Tabelle beschreibt die **Methoden** von DccConnection:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Delete	-	-	Löscht eine Verbindung.	-

5.25.4.7 DccImportOptions

DccImportOptions

Für den Import von Plänen sind diese Einstellungen möglich:

- DccImportOptions.None:
Der Import ist erfolgreich abgeschlossen, wenn es keinen Namenskonflikt gibt. Ansonsten wird die Ausnahme DccImportChartWithSameNameAlreadyAvailableException ausgegeben.
- DccImportOptions.RenameOnConflict:
Wenn es einen Namenskonflikt bei den Plänen gibt, erhält der neu importierte Plan einen automatisch generierten Namen durch das CFC. Falls z. B. der Plan mit dem Namen CFC_1 während des Imports bereits vorhanden ist, erhält der neu importierte Plan den Namen CFC_2.

Siehe auch

DriveControlChartComposition (Seite 1212)

5.25.4.8 DccChartPartitionComposition

DccChartPartitionComposition Features

DccChartPartitionComposition entspricht einem Set von Partitionen in DriveControlChart.

Die folgende Tabelle beschreibt die **Methoden** von DccChartPartitionComposition:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Create	DccChartPartition	string name	Erstellt eine neue Partition im Plan mit dem angegebenen Namen.	DccTextTooLongException, DccNameAlreadyUsedException, DccTooManyPartitionsException
Find	DccChartPartition	string name	Findet eine Partition anhand des Namens.	-

5.25.4.9 DccImportResultData

DriveParameter

DccImportResultData enthält Informationen zum Ergebnis eines Imports von Plänen.

Die folgende Tabelle beschreibt die **Attribute** von DccImportResultData:

Attributname	Datentyp	Schreibbar	Beschreibung
RemappedParameterNumbers	IDictionary<UInt16, UInt16>	Nur lesen	Während eines Imports kann es passieren, dass ein zu importierender DCC-Parameter bereits vorhanden ist. In diesem Fall erhält der zu importierende Parameter eine neue Nummer. Dieses Dictionary enthält dann alle neu zugeordneten Parameter. Der Schlüsselwert ist die vorherige Parameternummer, als sie exportiert wurde und der Wert ist der neu erstellte Parameter.

Siehe auch

DriveControlChartComposition (Seite 1212)

5.25.4.10 DccParameter

DccParameter

DccParameter entspricht einem Parameter, der in einem DCC-Plan auf einem DCC-Pin veröffentlicht ist.

Die folgende Tabelle beschreibt die **Attribute** von DccParameter:

Attributname	Datentyp	Schreibbar	Beschreibung	Exceptions
Number	ushort	Schreibbar	Die Parameternummer des DCC-Parameters.	DccParameterNumberAlreadyExistsException, DccParameterNumberOutOfRangeException
ParameterText	string	Schreibbar	Der Parametertext des DCC-Parameters.	DccTextTooLongException
IsSink	boolean	Nur lesen	Zeigt an, ob der DCC-Parameter eine Senke oder Quelle ist.	
IsSignal	boolean	Schreibbar	Zeigt an, ob der Parameter ein Signalparameter ist.	

5.25.4.11 DccPin

DccPin Features

DccPin entspricht einem Pin in einer DCC-Block-Instanz Plan innerhalb eines DCC-Plans.

Die folgende Tabelle beschreibt die **Attribute** von DccPin:

Attributname	Datentyp	Schreibbar	Beschreibung	Exception
Name	string	Nur lesen	Name des Pins.	-
Comment	string	Schreibbar	Der Kommentar des Pins.	TextTooLongException
Value	object	Schreibbar	Der Wert des Pins.	DccInvalidValueException
IsInput	boolean	Nur lesen	True falls der Pin ein Eingang ist, ansonsten false.	-
Invisible	boolean	Schreibbar	Zeigt an, ob der Pin unsichtbar ist oder nicht.	-
Unit	string	Schreibbar	Die Einheit des Pins.	-
ForTest	boolean	Schreibbar	Zeigt an, ob der Pin zur Überwachung eingestellt ist oder nicht.	-

Die folgende Tabelle beschreibt die **Navigatoren** von DccPin:

Name	Datentyp	Schreibbar	Beschreibung
Connections	DccConnectionComposition	Nur lesen	Alle Verbindungen des Pins.
Parameter	DccParameter	Nur lesen	Die DCC-Parameter, die mit dem Pin veröffentlicht werden.

Die folgende Tabelle beschreibt die **Methoden** von DccPin:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Connect	DccConnection	DccPin partner	Erstellt eine Pin-zu-Pin-Verbindung mit dem angegebenen Partnerpin	DccIllegalAssignmentException
Publish	DccParameter	bool setAsSignal	Veröffentlicht den Pin als Parameter. Wenn setAsSignal true ist, wird der Pin als Signal veröffentlicht und die Parameternummer wird automatisch gesetzt.	DccPublishingStructAlarmIdException, DccNoMoreFreeParameterNumberInDriveObjectException
Publish	DccParameter	bool setAsSignal, ushort parameterNumber	Veröffentlich den Pin mit der angegebenen Parameternummer. Wenn setAsSignal true ist, wird der Pin als Signal veröffentlicht.	DccParameterNumberAlreadyExistsException, DccParameterNumberOutOfRangeException, DccPublishingStructAlarmIdException, DccNoMoreFreeParameterNumberInDriveObjectException
Unpublish	-	-	Macht die Veröffentlichung des Pins als Parameter rückgängig und bestehende DCC-Verbindungen werden gelöscht.	-

5.25.4.12 DccPinComposition

DccPinComposition Features

DccPinComposition entspricht einem Set von Pins in einer DccBlock-Instanz.

Die folgende Tabelle beschreibt die **Methoden** von DccPinComposition:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Find	DccPin	string name	Findet einen Pin anhand des Namens.	-

5.25.4.13 DriveControlChart

DriveControlChart Features

DriveControlChart entspricht einem Plan oder Sub-Plan.

Die folgende Tabelle beschreibt die **Attribute** von DriveControlChart:

Attributname	Datentyp	Schreibbar	Beschreibung	Exception
Name	string	Nur lesen	Name des Plans.	DccTextTooLongException, DccNameInvalidException, DccNameAlreadyUsedException
Comment	string	Schreibbar	Der Kommentar von DriveControlChart.	DccTextTooLongException
Partition	DccChartPartition	Schreibbar	Die Partition, an der sich DriveControlChart im übergeordneten Plan befindet.	-

Attributname	Datentyp	Schreibbar	Beschreibung	Exception
PositionX	int	Schreibbar	Die X-Koordinate des Sub-Plan innerhalb des übergeordneten Plans.	-
PositionY	int	Schreibbar	Die Y-Koordinate des Sub-Plan innerhalb des übergeordneten Plans.	-
HorizontalSheets	uint	Schreibbar	Die Anzahl horizontaler Seiten im Plan.	DccInvalidValueException
VerticalSheets	uint	Schreibbar	Die Anzahl vertikaler Seiten im Plan.	DccInvalidValueException
SheetWidth	int32	Nur lesen	Die Breite des Blattes für die aktuelle Seitengröße.	-
SheetHeight	int32	Nur lesen	Die Höhe des Blattes für die aktuelle Seitengröße.	-

Die folgende Tabelle beschreibt die **Navigatoren** von DriveControlChart:

Name	Datentyp	Schreibbar	Beschreibung
Blocks	DccBlockComposition	Nur lesen	Gibt alle Blöcke innerhalb eines Plans in unsortierter Reihenfolge zurück.
Subcharts	DriveControlChartComposition	Nur lesen	Gibt alle Sub-Pläne eines Plans in unsortierter Reihenfolge zurück.
Partitions	DccPartitionComposition	Nur lesen	Gibt jedes Statement (Sub-Pläne und Blöcke) eines Plans in unsortierter Reihenfolge zurück.

Die folgende Tabelle beschreibt die **Methoden** von DriveControlChart:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Export	-	string path	Beschreibung Methode: Exportiert ausschließlich diesen Plan. Beschreibung Parameter: path: Vollständiger Pfad der zu exportierenden Datei.	DccExportException (Seite 1227)
Find	Statement	string name bool deepSearch	Findet einen Sub-Plan oder Block innerhalb des derzeitigen Plans oder im untergeordneten Sub-Plan, wenn deepSearch auf true gesetzt ist.	-
Delete	-	-	Beschreibung Methode: Löscht den Plan.	-
GetRunSequence	IList<Statement>	-	Gibt die Ablaufreihenfolge des Statement innerhalb des aktuellen Plans zurück.	-
OptimizeRunSequence	-	-	Beschreibung Methode: Optimiert die Ablaufreihenfolge des Plans. Der Optimierungsmechanismus wird von CFC bereitgestellt.	-
ShowDccEditor	-	-	Öffnet den DCC-Editor im TIA Portal für den Plan, wenn im interaktivem Modus.	DccException, DccChartsProtectedException

Siehe auch

DriveControlChart Exceptions (Seite 1227)

DriveControlChartComposition (Seite 1212)

Pläne löschen (Seite 1222)

5.25.4.14 DriveControlChartContainer**DriveControlChartContainer**

DriveControlChartContainer ist ein Dienst, der den DCC-Plan-Container unter einem DriveObject repräsentiert, sodass er aus dem entsprechenden DriveObject abgerufen werden kann. Wenn das DriveObject nicht von einem unterstützten Gerät stammt, ist der Dienst nicht verfügbar.

Die folgende Tabelle beschreibt die **Navigatoren** von DriveControlChartContainer:

Name	Datentyp	Schreibbar	Beschreibung
Charts	DriveControlChartComposition	Nur lesen	Liest das Plan-Layout im Plan-Container aus.
DcbLibraries	DcbLibraryComposition	Nur lesen	Gibt alle DCB-Bibliotheken zurück, die in den Plänen verwendet werden.

5.25.4.15 DriveControlChartComposition**DriveControlChartComposition features**

DriveControlChartComposition zeigt eine Liste der verfügbaren Pläne.

Die folgende Tabelle beschreibt die **Methoden** von DriveControlChartComposition:

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Create	DriveControlChart (Seite 1210)	string name	Erstellt einen neuen Plan mit dem vorgegebenem Namen.	DccTextTooLongException, DccNameInvalidException, DccNameAlreadyUsedException, DccTooManyChartsException, DccNestingTooDeepException, DccException
Create	DriveControlChart (Seite 1210)	-	Erstellt einen neuen Plan mit einem automatisch generierten Namen.	DccTooManyChartsException, DccNestingTooDeepException, DccException

Name	Typ Rückgabewert	Parameter	Beschreibung	Exceptions
Import	DcclImportResultData (Seite 1208)	string path, DcclImportOptions (Seite 1207) importOptions	Beschreibung Methode: Importiert Pläne aus einer DCC-Exportdatei. Beschreibung Parameter: path: Vollständiger Pfad der zu importierenden Datei. importOptions: Beim Import verwendete Optionen, siehe DcclImportOptions (Seite 1207).	DcclImportException (Seite 1227)
Export	-	string path	Beschreibung Methode: Exportiert alle Pläne in eine DCC-Exportdatei. Beschreibung Parameter: path: Vollständiger Pfad der zu exportierenden Datei.	DccExportException (Seite 1227)
GetChartSequence	DriveControlChart (Seite 1210)	-	Liest alle Pläne im Container in der Ablauffolgenfolge aus.	-
Find	DriveControlChart (Seite 1210)	string name	Identifiziert einen Plan anhand seines Namens im Plan-Container.	-

Siehe auch

DriveControlChartComposition Exceptions (Seite 1227)

DcclImportResultData (Seite 1208)

DriveControlChart (Seite 1210)

DcclImportOptions (Seite 1207)

5.25.4.16 Import DCB extension library

ImportDCBextensionlibrary

Die folgende Tabelle beschreibt die **Methoden** von ImportDCBextensionlibrary:

Name	Rückgabewert	Parameter	Beschreibung
ImportDcbLibrary	-	string path	Argumente Methode: path: Der vollständige Pfad der Zip-Datei der DCB Extension-Bibliothek. Beschreibung Methode: Importiert eine DCB Extension-Bibliothek in die Projektbibliothek.

5.25.4.17 Statement

Statement

Ein Statement stellt ein DriveControlChart oder einen DCC-Block dar. Alle folgenden Attribute und Methoden werden vererbt.

Die folgende Tabelle beschreibt die **Attribute** von Statement:

Attributname	Datentyp	Schreibbar	Beschreibung	Exception
Name	string	Schreibbar	Der Name von Statement.	DccNameTooLongException DccNameInvalidException
Comment	string	Schreibbar	Der Kommentar von Statement.	DccNameTooLongException DccNameInvalidException
Partition	DccChartPartition	Schreibbar	Die Partition, in der sich Statement im übergeordneten Plan befindet.	-
PositionX	int	Schreibbar	Die X-Koordinate des Statement innerhalb des übergeordneten Plans.	-
PositionY	int	Schreibbar	Die Y-Koordinate des Statement innerhalb des übergeordneten Plans.	-

Hinweis

Ungültige Koordinaten für ein Statement - CFC positioniert automatisch

Wenn eine der Koordinaten (PositionX, PositionY) für das Statement einen ungültigen Wert erhält (z. B. PositionY = -20), wird das Statement von CFC automatisch positioniert. Infolgedessen zeigt der Wert des entsprechenden Attributs (z. B. PositionY) immer noch den ungültigen Wert an. Dieses Verhalten ist beabsichtigt. Damit können Sie feststellen, ob ein Statement automatisch positioniert wurde.

Die folgende Tabelle beschreibt die **Methoden** von Statement:

Name	Rückgabewert	Parameter	Beschreibung	Exceptions
SetAsPredecessor	-	string path	Setzt das Statement als den Vorgänger des nächsten Statement, das in den Plan eingefügt wird. Auf einen Quellplan angewendet, hat die Methode keinen Effekt.	-
MoveInRuntimeSequence	-	int index	Bewegt das Statement in der Ablaufreihenfolge an den angegebenen Index.	DccInvalidValueException

5.25.5 Codebeispiele

5.25.5.1 Allgemein

Die folgenden Codebeispiele beschreiben die grundsätzliche Vorgehensweise für verschiedene Anwendungsfälle. Der Code ist nicht zwingend vollständig und kompilierbar.

Hinweis

Leistung optimieren

Für eine optimale Leistung bei der Nutzung von Openness wird empfohlen, betroffene Editoren (z. B. den DCC-Editor für Pläne, die Parameterliste usw.) zu schließen, solange eine aktive TIA Portal-Instanz mit Benutzeroberfläche geöffnet ist.

5.25.5.2 Ablaufreihenfolge ändern

Das folgende Beispiel zeigt, wie Sie die Ablaufreihenfolge der Pläne ändern können.

Ablaufreihenfolge der Pläne ändern

```
DriveControlChart chart;  
IList<IStatement> runtimeSequence = chart.GetRunSequence();  
runtimeSequence[3].MoveInRuntimeSequence(0);
```

5.25.5.3 Ablaufreihenfolge optimieren

Das folgende Beispiel zeigt, wie Sie die Ablaufreihenfolge der Pläne optimieren können.

Ablaufreihenfolge der Pläne optimieren

```
try  
{  
    DriveControlChart chart;  
    ...  
    chart.OptimizeRunSequence();  
}  
catch (DccLicenseUnavailableException exc)  
{  
}
```

Siehe auch

DCC Openness (Seite 1203)

5.25.5.4 Auf Pläne zugreifen

Das folgende Beispiel zeigt, wie Sie auf Pläne zugreifen können.

Auf Pläne zugreifen

```
DriveControlChartComposition charts = ...
DriveControlChart firstChart = charts[0];

// or looping...
foreach(DriveControlChart chart in charts)
{
    ...
}
```

5.25.5.5 Alle Pläne exportieren

Das folgende Beispiel zeigt, wie Sie alle Pläne exportieren können.

Alle Pläne exportieren

```
try
{
    charts.Export(@"c:\Charts.dcc");
}
catch (DccExportException exc)
{
}
```

Siehe auch

DCC Openness (Seite 1203)

5.25.5.6 DCB Extension-Bibliothek importieren

Das folgende Beispiel zeigt, wie Sie eine DCB Extension-Bibliothek importieren können.

DCB Extension-Bibliothek importieren

```
try
{
    Project myProject;
    ...

    myProject.ProjectLibrary.ImportDcbLibrary(@"c:\GMCV5_1_sinamics5_1_(5.1.15)
.zip");
}
catch (DccImportException exc)
{
}
```

Siehe auch

DCC Openness (Seite 1203)

5.25.5.7 DCB-Bibliothek Info auslesen

Das folgende Beispiel zeigt, wie Sie die Informationen einer DCB-Bibliothek auslesen können.

DCB-Bibliothek Info auslesen

```
DriveControlChartContainer chartContainer = ...
DcbLibraryComposition dcbLibraries = chartContainer.DcbLibraries;
DcbLibraryComposition testLib = dcbLibraries.Find("TestLib");
System.Version libVersion = testLib[0].Version;
System.Console.WriteLine($"Major version: {version.Major}, Minor version:
{version.Minor});
```

5.25.5.8 DCC-Blöcke bearbeiten

Das folgende Beispiel zeigt, wie Sie einen Block im Plan erstellen können.

Block im Plan erstellen

```
DriveControlChart chart;
var blocks = chart.Blocks;
DccBlock newBlock = blocks.Create("ADD")
DccBlock newBlock = blocks.Create("add_1", "ADD");
DccBlock newBlock = blocks.Create("add_1", "gear", "GMC");
```

Das folgende Beispiel zeigt, wie Sie einen Block in der unteren rechten Ecke des Plans platzieren können.

Block in der unteren rechten Ecke des Plans platzieren

```
DriveControlChartContainer chartContainer = ...
DriveControlChart chart = chartContainer.Charts.Find("DCC_1");
DccBlockComposition blocks = chart.Blocks;
DccBlock block = blocks.Create("ADD_1");

int horizontalSheetCount = chart.HorizontalSheets;
int verticalSheetCount = chart.VerticalSheets;
int blockSizeX = 20;
int blockSizeY = 20;

int sheetWidth = chart.SheetWidth;
int sheetHeight = chart.SheetHeight;

block.PositionX = (sheetWidth * horizontalSheetCount) - blockSizeX;
block.PositionY = (sheetHeight * verticalSheetCount) - blockSizeY;
```

Das folgende Beispiel zeigt, wie Sie einen Block anhand des Namens finden können.

Block anhand des Namens finden

```
DriveControlChart chart;  
var blocks = chart.Blocks;  
DccBlock addBlock = blocks.Find("add_1");
```

Das folgende Beispiel zeigt, wie Sie die Attribute eines Blocks einstellen können.

Attribute eines Blocks einstellen

```
DccBlock dccBlock;  
dccBlock.Name = "add_3";  
dccBlock.Comment = "new comment";  
dccBlock.PositionX = 110;  
dccBlock.PositionY = 234;  
dccBlock.GenericInputsNumber = 4;
```

Das folgende Beispiel zeigt, wie Sie einen Block aus dem Plan löschen können.

Block aus dem Plan löschen

```
DccBlock dccBlock = chart.Blocks.Find("add_2");  
dccBlock.Delete();
```

Das folgende Beispiel zeigt, wie Sie ein Statement (z. B. DCC-Block) als Vorgänger in der Ablaufreihenfolge setzen können.

Ein Statement (z. B. DCC-Block) als Vorgänger in der Ablaufreihenfolge setzen

```
DccBlock dccBlock;  
dccBlock.SetAsPredecessor();
```

5.25.5.9 DCB-Blocktyp Info auslesen

Das folgende Beispiel zeigt, wie Sie die Informationen eines DCB-Blocktyp auslesen können.

DCB-Blocktyp Info auslesen

```
DriveControlChartContainer chartContainer = ...  
DcbLibraryComposition dcbLibraries = chartContainer.DcbLibraries;  
DcbLibraryComposition testLib = dcbLibraries.Find("TestLib");  
DcbBlockTypeComposition blockTypes = testLib.DcbBlockTypes;  
DcbBlockType blockType = blockTypes.Find("test_all_pin_types");  
  
System.Console.WriteLine($"Name: {blockType.Name}, Description:  
{blockType.Description});
```

5.25.5.10 DCC-Parameter bearbeiten

Das folgende Beispiel zeigt, wie Sie die DCC-Parameter bearbeiten können.

DCC-Parameter ändern

```
DccPin pinOnControlUnit;  
DccParameter param1 = pinOnControlUnit.Parameter;  
param1.Number = 21600;  
param1.ParameterText = "new text for parameter";
```

5.25.5.11 Importbericht anzeigen

Das folgende Beispiel zeigt, wie Sie den Bericht zu einem Importvorgang anzeigen können.

Importbericht anzeigen

```
DccImportResultData importResultData = charts.Import(@"d:\Charts.dcc",  
DccImportOptions.None);  
IDictionary<ushort, ushort> remappedParameters = importResultData.RemappedP  
arameterNumbers;  
foreach (KeyValuePair<ushort, ushort> remappedParameter in remappedParamete  
rs)  
{  
    System.Console.WriteLine($"ParameterNumber <{remappedParameter.Key}> has  
been changed to <{remappedParameter.Value}> during import.");  
}
```

5.25.5.12 Pins bearbeiten

Das folgende Beispiel zeigt, wie Sie Pins bearbeiten können.

Pins anhand des Namens finden

```
DccBlock block;  
DccPinComposition pins = block.Pins;  
DccPin x3 = pins.Find("X3");
```

Attribute des Pins einstellen

```
DccPin dccPin;  
dccPin.Comment = "new comment";  
dccPin.Value = 42;
```

Pins verbinden

Pins verbinden

```
DccBlock block1;  
DccPin input = block.Pins.Find("X1");  
DccPin output = block.Pins.Find("Y");  
input.Connect(output);
```

Pins als Parameter veröffentlichen

```
DccBlock block1;  
DccPin input = block1.Pins.Find("X1");  
DccPin output = block1.Pins.Find("Y");  
input.Publish(true);  
output.Publish(false, 21620);
```

Veröffentlichung eines Pins als Parameter aufheben

```
output.Unpublish();
```

5.25.5.13 Pläne in der Ablaufreihenfolge abrufen

Das folgende Beispiel zeigt, wie Sie Pläne in der Ablaufreihenfolge abrufen können.

Pläne abrufen

```
DriveControlChartComposition charts = ...  
IList<DriveControlChart> chartSequence = charts.GetChartSequence();
```

5.25.5.14 Pläne oder Blöcke anhand des Namens finden

Das folgende Beispiel zeigt, wie Sie Pläne oder Blöcke anhand des Namens suchen können. Bei der Suche nach Plänen oder Blöcken macht bei den Suchbegriffen die Groß- und Kleinschreibung keinen Unterschied.

Einen Plan oder Block finden

```
DriveControlChartComposition charts = ...  
DriveControlChart chart = charts.Find("CFC_1");  
chart.Blocks.Find("add_1");
```

Siehe auch

DCC Openness (Seite 1203)

5.25.5.15 Plan oder Sub-Plan erstellen

Das folgende Beispiel zeigt, wie Sie einen neuen Plan oder Sub-Plan erstellen können.

DCC Plan oder Sub-Plan erstellen

```
DriveControlChartComposition charts = ...
DriveControlChart chart1 = charts.Create("DCC_1");
DriveControlChart chart2 = charts.Create(); // the next available number
will be used, so here it will be "DCC_2"
```

5.25.5.16 Pläne bearbeiten

Das folgenden Beispiele zeigen, wie Sie Pläne bearbeiten können.

Attribute eines Plans einstellen

```
DriveControlChart dccChart;
dccChart.Name = "DCC_3";
dccChart.Comment = "new comment";

// setting layout
dccChart.HorizontalSheets = 4;
dccChart.VerticalSheets = 3;

// applicable for a subchart
dccChart.Partition = parentChart.Partitions[1];
dccChart.PositionX = 110;
dccChart.PositionY = 234;
```

Ablaufreihenfolge auslesen und ändern

```
DriveControlChart chart;
IList<IStatement> runtimeSequence = chart.GetRunSequence();
runtimeSequence[3].MoveInRuntimeSequence(1);
```

Plan als Vorgänger setzen

```
DriveControlChart chart;
...
chart.SetAsPredecessor(); // if the chart is a root chart where the method
is called, then there is no effect as root chart cannot be predecessor in
the runtime sequence
```

DCC-Editor öffnen wenn TIA Portal im interaktiven Modus ist

DCC-Editor öffnen wenn TIA Portal im interaktiven Modus ist

```
DriveControlChart chart;
...
chart.ShowDccEditor();
```

DCC-Editor öffnen wenn TIA Portal im interaktiven Modus ist

```
DriveControlChart chart;
...
chart.ShowDccEditor();
```

Partitionen setzen

```
DriveControlChart chart;
...DccChartPartitionComposititionpartitions=chart.Partitions;
DccChartPartitionpartition2=partitions.Create("Partition_2");

// moving a block to the new partition
chart.Blocks.Find("add_2").Partition=partition2;
```

5.25.5.17 Pläne löschen

Das folgende Beispiel zeigt, wie Sie Pläne löschen können.

Pläne löschen

```
try
{
    DriveControlChart chart;
    ...
    chart.Delete();
}
{
}
```

Siehe auch

DCC Openness (Seite 1203)

DriveControlChart (Seite 1210)

5.25.5.18 Pläne abrufen

Das folgende Beispiel zeigt, wie Sie Pläne abrufen können.

Pläne abrufen

Pläne abrufen

```
DriveControlChartComposition charts = chartContainer.Charts;
```

5.25.5.19 Pläne importieren

Das folgende Beispiel zeigt, wie Sie Pläne importieren.

Pläne importieren

```
try
{
    DriveControlChartComposition charts = ...
    DccImportResultData result = charts.Import(@"c:\Charts.dcc",
    DccImportOptions.None);
}
catch (DccImportException exc)
{
}
```

Siehe auch

DCC Openness (Seite 1203)

5.25.5.20 Pläne exportieren

Das folgende Beispiel zeigt, wie Sie Pläne exportieren können.

Pläne exportieren

```
try
{
    DriveControlChart chart;
    ...
    chart.Export(@"c:\CFC_1.dcc");
}
catch (DccExportException exc)
{
}
```

Siehe auch

DCC Openness (Seite 1203)

5.25.5.21 Verbindungen bearbeiten

Das folgende Beispiel zeigt, wie Sie die Senke/Quelle einer Verbindung auslesen können.

Senke/Quelle einer Verbindung auslesen

Senke/Quelle einer Verbindung auslesen

```
DccConnection connection;  
var sink = connection.Sink;  
if (sink is DccParameter)  
{  
    System.Console.WriteLine("Sink is DCC parameter: " +  
        ((DccParameter)sink).ParameterNumber);  
}
```

5.25.5.22 Zugriff auf DriveControlChartContainer über DriveObject

Das folgende Beispiel zeigt, wie Sie über ein Antriebsobjekt auf DriveControlChartContainer zugreifen.

Zugriff auf DriveControlChartContainer

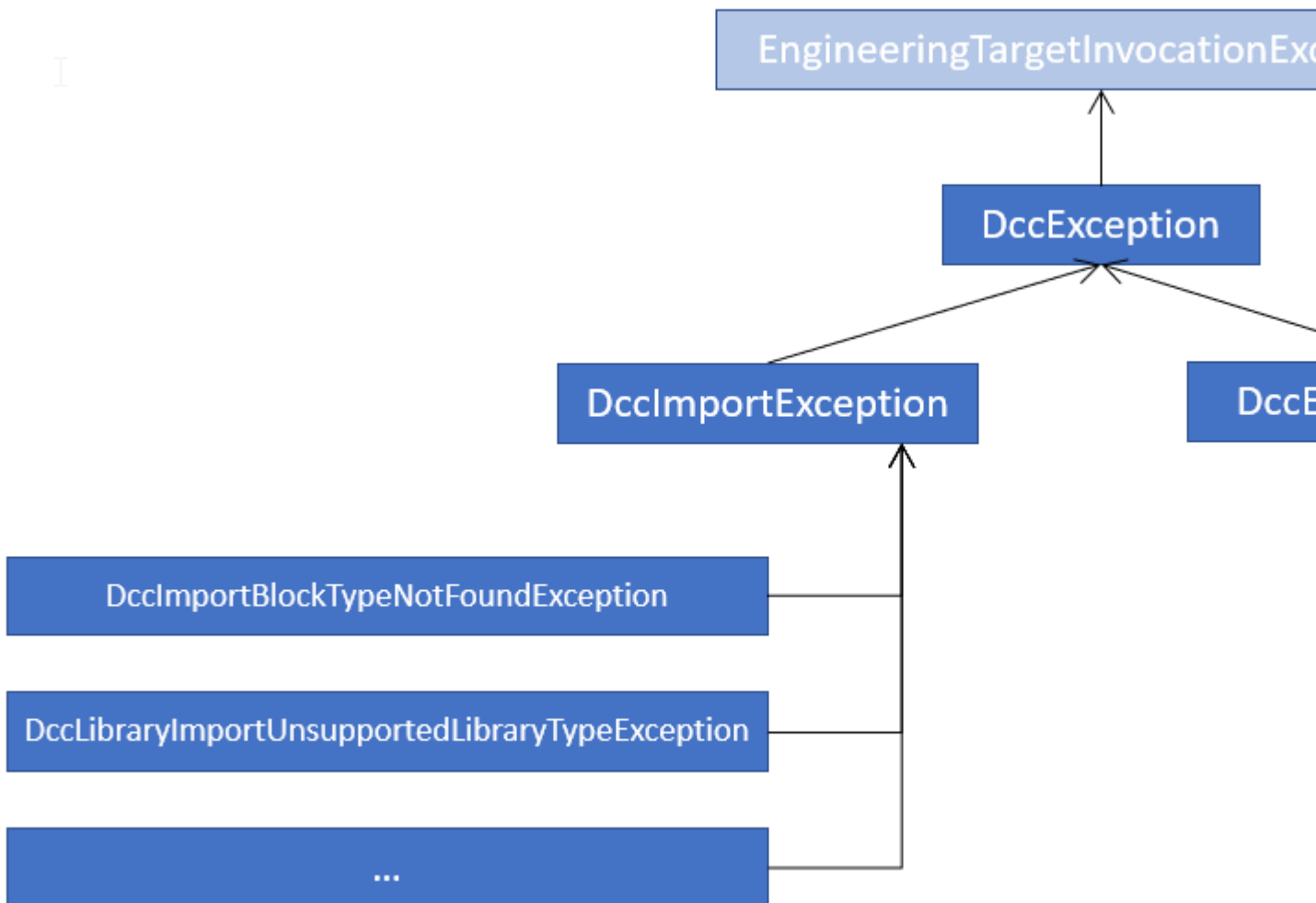
```
using Siemens.Engineering.MC.Drives;  
using Siemens.Engineering.MC.Drives.DCC;  
  
Project project = portal.Projects.Open("..."); //Destination folder to  
open the project  
DeviceItem item = project.Devices[0].Items[0];  
DriveObject  
driveObject = item.GetService<DriveObjectContainer>().DriveObjects[0];  
DriveControlChartContainer  
chartContainer = driveObject.GetService<DriveControlChartContainer>();  
//chartContainer can be null in case the DriveObject does not support DCC.
```

5.25.6 Ausnahmen DCC Openness**5.25.6.1 Umgang mit Exceptions****Umgang mit Exceptions**

Für Anwendungen, die aufgrund von Benutzerfehlern fehlschlagen könnten, löst DCC Openness eine DCC-spezifische Ausnahme aus, die von `EngineeringTargetInvocationException` abgeleitet wird. Die Ausnahmen sind hierarchisch aufgebaut, wodurch alle spezifischen Ausnahmen von der übergeordneten Ausnahme `DccException` abgeleitet werden:

Hinweis**Exception bei fehlender Lizenz**

Die Exception `DccLicenseUnavailableException` wird ausgegeben, wenn die notwendige DCC-Lizenz für eine Funktion nicht vorhanden ist.



Wenn Sie nicht im Detail unterschiedliche Fehlerursachen untersuchen oder auswerten möchten, ist der einfachste Weg, die allgemeine `DccException` abzufangen:

```

try
{
    Project myProject;
    ...

    myProject.ProjectLibrary.ImportDcbLibrary(@"c:\GMCV5_1_sinamics5_1_(5.1.15)
.zip");
    ...
    DriveControlChartContainer chartContainer = driveObject.GetService<DriveC
ontrolChartContainer>();
    chartContainer.Charts.Import(@"d:\Charts.dcc", DccImportOptions.None);
}
catch (DccException exc)
{
}

```

Fangen Sie alle relevanten Ausnahmen ab, wenn Sie je nach Fehlerart unterschiedlich reagieren möchten:

```

try
{
    DriveControlChartContainer
chartContainer = driveObject.GetService<DriveControlChartContainer>();
    chartContainer.Charts.Import(@"d:\Charts.dcc", DccImportOptions.None);
}
catch (DccImportLibraryIsMissingException missingLibExc)
{
}
catch (DccImportChartWithSameNameAlreadyAvailableException sameNameExc)
{
}
catch (DccImportException exc)
{
}
catch (DccException exc)
{
}

```

5.25.6.2 DccBlockComposition Exceptions

Ausnahmen

Ausnahmen, die bei der Erstellung eines Blocks ausgegeben werden können:

- **DccInvalidNameException:**
Wird ausgegeben, wenn der Namen des Blocks ungültig ist.
- **DccTooLongTextException:**
Wird ausgegeben, wenn der Name des Blocks zu lang ist.

- **DccNameAlreadyUsedException:**
Wird ausgegeben, wenn der Name des Blocks bereits im Plan verwendet wird.
- **DccBlocktypeNotFoundException:**
Wird ausgegeben, wenn der angegebene Blocktyp in keiner der vorhandenen Bibliotheken gefunden werden kann.

5.25.6.3 DriveControlChart Exceptions

Ausnahmen

Ausnahmen, die bei einem Export ausgegeben werden können:

- **DccExportException:**
Wird bei jeder Art von Exportfehler ausgegeben.

Ausnahmen, die beim ändern der Ablaufreihenfolge ausgegeben werden können:

- **DccException:**
Wird ausgegeben, wenn die Ablaufreihenfolge nicht geändert werden kann.

Ausnahmen, die beim Ausführen der Optimierungsfunktion für die Ablaufreihenfolge von Plänen ausgegeben werden können:

- **DccLicenseUnavailableException:**
Wird ausgegeben, wenn keine DCC-Lizenz vorhanden ist. In diesem Fall steht die Optimierungsfunktion für die Ablaufreihenfolge nicht zur Verfügung.

Siehe auch

DriveControlChart (Seite 1210)

5.25.6.4 DriveControlChartComposition Exceptions

Ausnahmen

Ausnahmen, die beim Ausführen von Import ausgegeben werden:

- **DccException:**
Wird ausgegeben, wenn ein neuer Plan nicht erstellt werden kann.
Wird ausgegeben, wenn die Ablaufreihenfolge von Plänen nicht geändert werden kann.
- **DccTextTooLongException:**
Wird ausgegeben, wenn ein neuer Plan nicht erstellt werden kann weil der Name zu lang ist.
- **DccNameInvalidException:**
Wird ausgegeben, wenn ein neuer Plan nicht erstellt werden kann weil der Name ungültig ist (z. B. weil der Name unzulässige Zeichen enthält).
- **DccNameAlreadyUsedException:**

- **DccTooManyChartsException:**
Wird ausgegeben, wenn die maximale Anzahl von Plänen überschritten wurde.
- **DccNestingTooDeepException:**
Wird ausgegeben, wenn die maximale Anzahl ineinander verschachtelter Sub-Pläne überschritten wurde.
- **DccImportException:**
Allgemeine Importausnahme, die DCC im Fall eines Importfehlers ausgibt.
- **DccImportBlockCreationException:**
Wird ausgegeben, wenn ein Block während eines Imports nicht erstellt werden kann.
- **DccImportBlockTypeNotFoundException:**
Wird ausgegeben, wenn ein referenzierter DCB-Blocktyp weder in einer Standard-DCBLib oder einer Extension-Bibliothek gefunden werden kann.
- **DccImportChartCreationException:**
Wird ausgegeben, wenn ein Plan während eines Imports nicht erstellt werden kann.
- **DccImportChartWithSameNameAlreadyAvailableException:**
Wird ausgegeben, wenn ein Plan mit demselben Namen bereits im Plan-Container vorhanden ist und DccImportOptions.None verwendet wird.
- **DccImportLibraryIsMissingException:**
Wird ausgegeben, wenn eine referenzierte DCB Extension-Bibliothek nicht im Projekt vorhanden ist.
- **DccImportFileAlreadyInUseException:**
Wird ausgegeben, wenn eine zu importierende Datei bereits durch einen anderen Prozess verwendet wird.
- **DccImportDcbTypeDifferentVersionAlreadyUsedException:**
Wird ausgegeben, wenn ein DCB-Typ in einer anderen Version der gleichen DCB Extension-Bibliothek bereits im Gerät verwendet wird.

Siehe auch

DriveControlChartComposition (Seite 1212)

5.25.6.5 ImportDcbLibrary Exceptions

Ausnahmen

Ausnahmen, die von ImportDcbLibrary ausgegeben werden:

- **DccImportException:**
Wird bei jeder Art von Importfehler ausgegeben.
- **DccLibraryImportAlreadyAvailableException:**
Wird ausgegeben, wenn bereits eine DCB Extension-Bibliothek mit der gleichen Version im Projekt vorhanden ist.

- `DccLibraryImportCorruptedStudioLibraryException`:
Wird ausgegeben, wenn die Zip-Datei der DCB Extension-Bibliothek kompromittiert ist und nicht importiert werden kann.
- `DccLibraryImportIntegrityBrokenException`:
Wird ausgegeben, wenn die Integrität der Zip-Datei der DCB Extension-Bibliothek kompromittiert ist.
- `DccLibraryImportOverallPinLimitExceededException`:
Wird ausgegeben, wenn die DCB Extension-Bibliothek einen DCB-Typ enthält, dessen Pinanzahl die maximal Pinanzahl überschreitet.
- `DccLibraryImportStandardLibraryAlreadyAvailableException`:
Wird ausgegeben, wenn die zu importierende ausgewählte DCB Extension-Bibliothek die Standard-DCBLib ist, die bereits vorhanden ist.
- `DccLibraryImportUnsupportedLibraryTypeException`:
Wird ausgegeben, wenn die ausgewählte DCB Extension-Bibliothek nicht durch DCC und SINAMICS-Antriebe unterstützt wird.

5.26 Ausnahmen

5.26.1 Umgang mit Exceptions

Ausnahmen beim Zugriff auf das TIA Portal über TIA Portal Openness APIs

Bei der Ausführung einer TIA Portal Openness-Anwendung über die TIA Portal Openness API werden alle auftretenden Fehler als Ausnahmen gemeldet. Diese Ausnahmen enthalten Informationen, die Ihnen helfen, die aufgetretenen Fehler zu beheben.

Dabei wird zwischen zwei Arten von Ausnahmen unterschieden:

- **Recoverable** (Siemens.Engineering.EngineeringException)
Sie können mit dieser Ausnahme weiterhin ohne Unterbrechung auf das TIA Portal zugreifen. Alternativ können Sie die Verbindung zum TIA Portal abbrechen.
Die EngineeringExceptions beinhalten die folgenden Typen:
 - Sicherheitsbezogene Ausnahmen (EngineeringSecurityException), beispielsweise bei fehlenden Zugriffsrechten.
 - Ausnahmen beim Zugriff auf Objekte (EngineeringObjectDisposedException), beispielsweise beim Zugriff auf nicht mehr vorhandene Objekte.
 - Ausnahmen beim Zugriff auf Attribute (EngineeringNotSupportedException), beispielsweise beim Zugriff auf nicht mehr vorhandene Attribute.
 - Allgemeine Ausnahmen beim Aufruf (EngineeringTargetInvocationException), beispielsweise bei einem Fehler trotz gültigen Aufrufs der TIA Portal Openness API.
 - Ausnahmen beim Aufruf (EngineeringRuntimeException), beispielsweise bei einer ungültigen Belegung.
 - Ausnahmen, wenn es nicht genügend Ressourcen in der zugeordneten TIA Portal-Instanz gibt (EngineeringOutOfMemoryException)
 - Ausnahmen bei beendeten Aufrufen (EngineeringUserAbortException), beispielsweise beim Abbruch des Importvorgangs durch den Benutzer.
 - Ausnahmen beim API-Aufruf durch einen vom Kunden bereitgestellten Delegate (EngineeringDelegateInvocationException). Diese Ausnahme wird von der Ausnahme EngineeringTargetInvocationException abgeleitet.

Die EngineeringExceptions haben die folgenden Attribute:

- `ExceptionMessageData messageData`: Enthält den Grund, weshalb die Ausnahme ausgelöst wurde.
- `ExceptionMessageData detailMessageData`: Enthält zusätzliche Informationen über den Grund. Das Ergebnis wird als `<IList>` zurückgegeben.
- `String message`: Gibt das Ergebnis von `MessageData` und `DetailMessageData` zurück.

`ExceptionMessageData` gibt die folgenden Informationen zurück:

- `String Text`: Enthält den Grund, weshalb die Ausnahme ausgelöst wurde.
- **NonRecoverable** (Siemens.Engineering.NonRecoverableException)
Diese Ausnahme schließt das TIA Portal und die Verbindung zum TIA Portal wird getrennt. Sie müssen das TIA Portal mit der TIA Portal Openness-Anwendung erneut starten.

Programmcode

Das folgende Beispiel zeigt die Möglichkeiten, die Sie haben, um auf Ausnahmen zu reagieren:

```
try
{
    ...
}

catch(EngineeringSecurityException engineeringSecurityException)
{
    Console.WriteLine(engineeringSecurityException);
}

catch(EngineeringObjectDisposedException engineeringObjectDisposedException)
{
    Console.WriteLine(engineeringObjectDisposedException.Message);
}

catch(EngineeringNotSupportedException engineeringNotSupportedException)
{
    Console.WriteLine(engineeringNotSupportedException.MessageData.Text);
    Console.WriteLine();
    foreach(ExceptionMessageData detailMessageData in
engineeringNotSupportedException.DetailMessageData)
    {
        Console.WriteLine(detailMessageData.Text);
    }
}

catch (EngineeringTargetInvocationException)
{
    throw;
}

catch (EngineeringException)
{
    //Do not catch general exceptions
    throw;
}

catch(NonRecoverableException nonRecoverableException)
{
    Console.WriteLine(nonRecoverableException.Message);
}
```

5.27 F-spezifisches Openness

5.27.1 F-spezifisches Openness

Voraussetzung

Änderungen im F-Programm können nur im Offline-Betrieb durchgeführt werden. In allen anderen Fällen wird eine Ausnahme erzeugt.

Wenn ein Passwort für ein F-Programm eingerichtet wurde, muss der Benutzer angemeldet sein, um Änderungen im F-Programm ausführen zu können. Andernfalls wird der Benutzer mit einer Ausnahme aufgefordert, sich zuerst im F-Programm anzumelden.

Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden.

Siehe "Verbindung zum TIA Portal aufbauen (Seite 84)".

Openness-Dienst

Die Openness-Schnittstelle (`Siemens.Engineering.dll`) ist um den Dienst `GlobalSettings` erweitert (siehe Namensbereich `Siemens.Engineering.Safety`), der zwei Aktionen ermöglicht:

- `SafetyModificationsPossible(bool safetyModificationsPossible)`
- `UsernameForFChangeHistory(string userName)`

Die Openness-Schnittstelle (`Siemens.Engineering.dll`) ist um den Dienst `SafetySignatureProvider` erweitert, der drei Aktionen ermöglicht:

- `SafetySignatureComposition Signatures`
- `UInt64 SafetySignature.Value`
- `SafetySignatureType SafetySignature.Type`

Die Openness-Schnittstelle (`Siemens.Engineering.dll`) ist um den Dienst `SafetyAdministration` mit den folgenden Aktionen und Eigenschaften im Namensraum `Siemens.Engineering.Safety` erweitert.

- `bool IsSafetyOfflineProgramPasswordSet`
- `void SetSafetyOfflineProgramPassword(SecureString newPassword)`
- `void RevokeSafetyOfflineProgramPassword(SecureString currentPassword)`
- `bool IsLoggedInToSafetyOfflineProgram`
- `void LoginToSafetyOfflineProgram(SecureString currentPassword)`
- `void LogoffFromSafetyOfflineProgram()`

Die Openness-Schnittstelle (`Siemens.Engineering.dll`) ist um den Dienst `SafetyAdministration` erweitert, auf den über ein `Plc-DeviceItem` zugegriffen werden kann. `SafetyAdministration` stellt die folgenden Eigenschaften bereit, die im

Namensraum **Siemens.Engineering.Safety** festgelegt sind. Diese beiden Eigenschaften enthalten weitere Eigenschaften, die im folgenden Abschnitt beschrieben werden.

- RuntimegroupComposition RuntimeGroups { get; }
- SafetySettings Settings { get; }

Die Openness-Schnittstelle (Siemens.Engineering.dll) ist um den Dienst `SafetyPrintout` erweitert, der mit der folgenden Aktion im Namensraum `Siemens.Engineering.Safety` **verknüpft ist**.

- `bool Print(SafetyPrintoutFilePrinter filePrinter, FileInfo fullOutputPath, string documentLayout, SafetyPrintoutOption documentationOption);`

Prinzip

Der Dienst `Safety.GlobalSettings` wird aus der `TiaPortal`-Instanz wie folgt aufgerufen:
`Engineering.Safety.GlobalSettings globalSettings =`
`TiaPortal.GetService<Engineering.Safety.GlobalSettings>();`

5.27.2 SafetyModificationsPossible

Anwendung

Die Aktion `SafetyModificationsPossible(bool safetyModificationsPossible)` des Dienstes `GlobalSettings` dient dazu, Änderungen am Sicherheitsprogramm im TIA Portal zu verhindern.

Wenn der Parameter `safetyModificationsPossible` den Wert `true` hat, verhält sich das TIA Portal entsprechend den aktuellen Einstellungen des Sicherheitsprogramms.

Wenn der Parameter `safetyModificationsPossible` den Wert `false` hat, werden alle Änderungen am Sicherheitsprogramm verhindert, unabhängig davon, ob der Benutzer bereits das Passwort für Änderungen am Sicherheitsprogramm eingegeben hat. Das System gibt in diesem Fall Feedback mit dem Hinweis darauf, dass der aktuelle Benutzer nicht berechtigt ist, das Sicherheitsprogramm zu ändern.

Wenn für das Sicherheitsprogramm kein Passwort konfiguriert oder vergeben wurde, hat der Wert `"false"` für `safetyModificationsPossible` keine Auswirkung. Dies bedeutet zugleich, dass das Sicherheitsprogramm geändert werden kann. Es ist jedoch nicht möglich, neue Passwörter einzurichten.

Die folgende Tabelle zeigt die von der Methode benötigten Parameter:

Name	Typ	Beschreibung
<code>safetyModificationsPossible</code>	<code>bool</code>	Autorisierung zu Änderungen am Sicherheitsprogramm

Programmcode

Verhindert Änderungen am Sicherheitsprogramm:

```
globalSettings.SafetyModificationsPossible(false);
```

5.27.3 UsernameForFChangeHistory

Anwendung

Mit der Aktion `UsernameForFChangeHistory(string userName)` wird der Benutzername angegeben, der vom TIA Portal für die anschließende Protokollierung in der F-Änderungshistorie verwendet wird.

Die maximale Länge der Zeichenkette beträgt 256 Zeichen. Zeichenketten, die die maximale Länge überschreiten, werden abgeschnitten.

Mit einem leeren Benutzernamen (null oder leere Zeichenkette) wird der Benutzername auf den Standardwert zurückgesetzt.

Die folgende Tabelle zeigt die von der Methode benötigten Parameter:

Name	Typ	Beschreibung
userName	String	Gewünschter Benutzername

Programmcode

Legt den gewünschten Benutzernamen fest:

```
globalSettings.UsernameForFChangeHistory("username");
```

5.27.4 SafetySignatureProvider

Das Auslesen der Signaturen gibt einen ersten Anhaltspunkt für mögliche Änderungen an F-Bausteinen. Zur Abnahme der Anlage fahren Sie fort wie im Kapitel "Abnahme der Anlage" des Handbuchs "SIMATIC Safety - Projektieren und Programmieren" beschrieben.

Anwendung

Signaturen: `SafetySignatureComposition` gibt eine Sammlung von `SafetySignature`-Objekten zurück. Objekte dieses Typs haben eine Eigenschaft `SafetySignature.Value: UInt64`, durch die für die F-Signatur ein ganzzahliger Wert zurückgegeben wird. Außerdem haben sie die Eigenschaft `SafetySignature.Type: SafetySignatureType`, durch die der Typ der F-Signatur zurückgegeben wird, der in der `SafetySignature`-Instanz eingekapselt ist.

Wenn die F-Signatur nicht vorhanden oder aufgrund einer kürzlichen Änderung nicht mehr gültig ist, wird 0 zurückgegeben. Dies gilt auch für Bausteine ohne F-Signatur (z. B. Systembausteine).

Programmcode

Der Dienst `SafetySignatureProvider` wird aus der `PlcBlock`-Instanz wie folgt aufgerufen:

```
SafetySignatureProvider signatureProvider =  
block.GetService<SafetySignatureProvider>()
```

Wenn der Baustein kein F-Baustein oder kein Baustein einer F-CPU S7-1200/1500 ist, ist der zurückgegebene `signatureProvider` null.

Bei einem Baustein, der nicht den Wert null annehmen kann, können Sie die F-Signatur mit folgendem Code abfragen:

```
SafetySignature signature =  
signatureProvider.Signatures.Find(SafetySignatureType.BlockOfflineSi  
gnature);  
UInt64 value = signature.value;
```

5.27.5 SafetyAdministration

5.27.5.1 SafetyAdministration

Diese Funktionen sind für S7-1200/1500 F-CPUs verfügbar.

Für Aktionen und die Einstellung von Werten muss eine gültige Safety-Lizenz installiert sein. Eine fehlende Lizenz verursacht eine Ausnahme. Das Lesen von Werten ist ohne Lizenz möglich.

Programmcode

Rufen Sie den Dienst `SafetyAdministration` über die verfügbare `PLC DeviceItem`-Instanz auf.

```
SafetyAdministration safetyAdministration =  
deviceItem.GetService<SafetyAdministration>();  
// check if the service is not null  
if (safetyAdministration != null)  
{  
    // ...  
}
```

5.27.5.2 IsSafetyOfflineProgramPasswordSet

Programmcode

```
bool isPasswordSet =  
safetyAdministration.IsSafetyOfflineProgramPasswordSet
```

Anwendung

Die Eigenschaft `IsSafetyOfflineProgramPasswordSet` gibt "true" zurück, wenn ein Passwort für das Sicherheitsprogramm festgelegt ist.

5.27.5.3 SetSafetyOfflineProgramPassword

Programmcode

```
if (!safetyAdministration.IsSafetyOfflineProgramPasswordSet)
{
    SecureString newPassword = new NetworkCredential("", "new
password").SecurePassword;

    safetyAdministration.SetSafetyOfflineProgramPassword(newPassword);
}
```

Anwendung

Mit der Aktion `SetSafetyOfflineProgramPassword` wird das Passwort für das Sicherheitsprogramm festgelegt, wenn es noch nicht festgelegt wurde. Nachdem das Passwort erfolgreich festgelegt wurde, ist der Benutzer nicht beim Sicherheitsprogramm angemeldet. Die Aktion `LoginToSafetyOfflineProgram` muss explizit aufgerufen werden, um eine Anmeldung auszuführen.

In folgenden Fällen wird dem Benutzer eine Ausnahme zurückgemeldet:

- Das angegebene Passwort ist ungültig (z. B. Länge gleich 0 oder größer als 30)
- Das angegebene Passwort ist null
- Ein Passwort für das Sicherheitsprogramm ist bereits definiert
- Wenn das Sicherheitsprogramm nicht geändert werden kann (siehe "SafetyModificationsPossible (Seite 1233)") oder das Projekt ist UMAC-geschützt, ist die Änderung des Passworts für das Sicherheitsprogramm über Openness nicht zulässig.
- Der Benutzer ist nicht offline
- Eine Safety-Lizenz ist nicht vorhanden

5.27.5.4 RevokeSafetyOfflineProgramPassword

Verwendung

Mit der Aktion `RevokeSafetyOfflineProgramPassword` wird das Passwort für das F-Programm gelöscht, wenn es korrekt angegeben wurde.

Nach erfolgreicher Aufhebung des Passworts kann der Benutzer mit der Aktion `SetSafetyOfflineProgramPassword` ein neues gültiges F-Programmpasswort festlegen.

In folgenden Fällen wird dem Benutzer eine Ausnahme zurückgemeldet:

- Wenn kein Passwort festgelegt ist, kann es nicht aufgehoben werden
- Das angegebene Passwort ist null
- Wenn das Sicherheitsprogramm nicht geändert werden kann (siehe "SafetyModificationsPossible (Seite 1233)") oder das Projekt ist UMAC-geschützt, ist die Änderung des Passworts für das Sicherheitsprogramm über Openness nicht zulässig.
- Der Benutzer ist nicht offline
- Eine Safety-Lizenz ist nicht vorhanden

Programmcode

```
if (safetyAdministration.IsSafetyOfflineProgramPasswordSet)
{
    SecureString password = new NetworkCredential("", "password").SecurePassword;
    safetyAdministration.RevokeSafetyOfflineProgramPassword(password);
}
```

5.27.5.5 IsLoggedInToSafetyOfflineProgram

Programmcode

```
bool isLoggedIn =
    safetyAdministration.IsLoggedInToSafetyOfflineProgram
```

Anwendung

Die Eigenschaft `IsLoggedInToSafetyOfflineProgram` gibt "true" zurück, wenn ein Passwort festgelegt wurde und der Benutzer angemeldet ist.

5.27.5.6 LoginToSafetyOfflineProgram

Programmcode

```
if (!safetyAdministration.IsLoggedInToSafetyOfflineProgram)
{
    SecureString password = new NetworkCredential("",
    "password").SecurePassword;
    safetyAdministration.LoginToSafetyOfflineProgram(password);
}
```

Anwendung

Mit der Aktion `LoginToSafetyOfflineProgram` wird der Benutzer beim Sicherheitsprogramm angemeldet, wenn das `Password` korrekt angegeben wurde. Nach einer erfolgreichen Anmeldung können Änderungen am Sicherheitsprogramm vorgenommen werden.

In folgenden Fällen wird dem Benutzer eine Ausnahme zurückgemeldet:

- Ein Passwort für das Sicherheitsprogramm ist nicht definiert
- Das Passwort ist null
- Das Passwort ist falsch
- Wenn das Sicherheitsprogramm nicht geändert werden kann (siehe "SafetyModificationsPossible (Seite 1233)") oder das Projekt ist UMAC-geschützt, ist die Änderung des Passworts für das Sicherheitsprogramm über Openness nicht zulässig.
- Der Benutzer ist nicht offline
- Eine Safety-Lizenz ist nicht vorhanden

5.27.5.7 LogoffFromSafetyOfflineProgram

Programmcode

```
if (safetyAdministration.IsLoggedInToSafetyOfflineProgram)
{
    safetyAdministration.LogoffFromSafetyOfflineProgram();
}
```

Anwendung

Mit der Aktion `LogoffFromSafetyOfflineProgram` wird der Benutzer vom Sicherheitsprogramm abgemeldet. Dadurch ist es nicht mehr möglich, Änderungen am Sicherheitsprogramm vorzunehmen.

In folgenden Fällen wird dem Benutzer eine Ausnahme zurückgemeldet:

- Ein Passwort für das Sicherheitsprogramm ist nicht definiert
- Wenn das Sicherheitsprogramm nicht geändert werden kann (siehe "SafetyModificationsPossible (Seite 1233)") oder das Projekt ist UMAC-geschützt, ist die Änderung des Passworts für das Sicherheitsprogramm über Openness nicht zulässig.
- Der Benutzer ist nicht offline
- Eine Safety-Lizenz ist nicht vorhanden

5.27.5.8 Ablaufgruppen

Eine neue Ablaufgruppe erstellen

Programmcode

```
// Erstellt eine Ablaufgruppe mit dem angegebenen Namen und
// erstellt einen Main-Safety-FB und einen Main-Safety-IDB.
RuntimeGroup RuntimeGroups.Create("F-runtime group 2");

// Erstellt eine Ablaufgruppe mit dem angegebenen FC für den Main-
// Safety-Block.
PlcBlock mainSafetyFC;
RuntimeGroup RuntimeGroups.Create("F-runtime group 2",
mainSafetyFC);

// Erstellt eine Ablaufgruppe mit dem angegebenen FB und IDB für
// den Main-Safety-Block.
PlcBlock mainSafetyFB;
PlcBlock mainSafetyIDB;
RuntimeGroup RuntimeGroups.Create("F-runtime group 2",
mainSafetyFB, mainSafetyIDB);
```

Verwendung

Ermöglicht, eine neue Ablaufgruppe mit den spezifizierten Parametern zu erstellen.

Eine Ausnahme wird erzeugt:

- wenn der angegebene Name ungültig ist.
- wenn bereits eine Ablaufgruppe mit dem angegebenen Namen existiert.
- wenn die höchstmögliche Anzahl von Ablaufgruppen überschritten wird.
- wenn die angegebenen Main-Safety-Blocks ungültig oder vom falschen Typ sind.

Eine Ablaufgruppe löschen

Programmcode

```
runtimeGroup.Delete();
```

Verwendung

Mit diesem Methodenaufruf wird die Ablaufgruppe gelöscht. Die referenzierten Bausteine in der Ablaufgruppe werden nicht gelöscht.

Ablaufgruppen

Programmcode

```
foreach (RuntimeGroup runtimeGroup in
safetyAdministration.RuntimeGroups)
{
    // access the properties on the RuntimeGroup object...
}

RuntimeGroup runtimeGroup1 =
safetyAdministration.RuntimeGroups.Find("RTG1");
```

Verwendung

Die Eigenschaft `RuntimeGroups` gibt alle Ablaufgruppen zurück. Mit Methode `Find` kann ferner nach einer bestimmten Ablaufgruppe gesucht werden. Die Methode `Find` gibt null zurück, wenn die Ablaufgruppe mit dem gesuchten Namen nicht existiert.

Auf die folgenden Eigenschaften kann von einer Instanz einer `RuntimeGroup` zugegriffen werden.

Name der Ablaufgruppe

Programmcode

```
string name = runtimeGroup.Name;
```

Verwendung

Gibt den Namen der Ablaufgruppe zurück.

F-OB Name

Programmcode

```
string name = runtimeGroup.FOBName;
```

Verwendung

Gibt den F-OB Namen zurück oder null, wenn kein F-OB eingestellt ist.

F-OB Ereignisklasse

Programmcode

```
string eventClass = runtimeGroup.FOBEventClass;
```

Verwendung

Gibt die F-OB Ereignisklasse zurück oder null, wenn kein F-OB festgelegt ist.

F-OB Nummer

Programmcode

```
Int32 number = (Int32) runtimeGroup.GetAttribute("FOBNumber");  
runtimeGroup.SetAttribute("FOBNumber", 123);
```

Verwendung

Gibt die F-OB Nummer zurück bzw. legt diese fest. Ist die F-OB Nummer nicht verfügbar, wird null zurückgegeben. Wird der Wert gesetzt und ist kein F-OB verfügbar, oder wenn der Wert sich nicht im gültigen Bereich befindet, wird eine Ausnahme erzeugt.

F-OB Zykluszeit

Programmcode

```
Int32 cycleTime = (Int32)  
runtimeGroup.GetAttribute("FOBCycleTime");  
runtimeGroup.SetAttribute("FOBCycleTime", 100000);
```

Verwendung

Gibt die Zykluszeit der Ablaufgruppe zurück bzw. legt diese fest. Ist der Wert nicht verfügbar, wird null zurückgegeben. Ist der gesetzte Wert nicht verfügbar oder nicht im gültigen Bereich, wird eine Ausnahme erzeugt.

F-OB Phasenverschiebung

Programmcode

```
Int32 phaseShift = (Int32)  
runtimeGroup.GetAttribute("FOBPhaseShift");  
runtimeGroup.SetAttribute("FOBPhaseShift", 120);
```

Verwendung

Gibt die Phasenverschiebung des fehlersicheren Organisationsbausteins der Ablaufgruppe zurück bzw. legt diese fest. Ist der Wert nicht verfügbar, wird null zurückgegeben. Wenn der gesetzte Wert nicht verfügbar ist oder nicht im gültigen Bereich liegt, wird eine Ausnahme erzeugt.

F-OB Priorität

Programmcode

```
Int32 priority = (Int32) runtimeGroup.GetAttribute("FOBPRIORITY");  
runtimeGroup.SetAttribute("FOBPRIORITY", 10);
```

Verwendung

Gibt die Priorität des fehlersicheren Organisationsbausteins der Ablaufgruppe zurück bzw. legt diese fest. Ist der Wert nicht verfügbar, wird null zurückgegeben. Wenn der gesetzte Wert nicht verfügbar ist oder nicht im gültigen Bereich liegt, wird eine Ausnahme erzeugt.

Name des Main-Safety-Blocks

Programmcode

```
string mainSafetyName = runtimeGroup.MainSafetyBlockName;
```

Verwendung

Gibt den Namen des Main-Safety-Blocks zurück bzw. null, wenn kein Baustein eingerichtet ist.

IDB-Name des Main-Safety-Blocks

Programmcode

```
string mainSafetyIDbName = runtimeGroup.MainSafetyBlockIDbName;
```

Verwendung

Gibt den Namen des Main-Safety-Instanz-DB zurück bzw. null, wenn kein Baustein eingerichtet ist.

Warnzykluszeit

Programmcode

```
Int32 warnCycleTime = runtimeGroup.WarnCycleTime;  
runtimeGroup.WarnCycleTime = 110000;
```

Verwendung

Gibt die Warnzykluszeit der Runtime-Gruppe zurück bzw. legt diese fest. Liegt der festgelegte Wert nicht im gültigen Bereich, wird eine Ausnahme erzeugt.

Maximale Zykluszeit

Programmcode

```
Int32 maximumCycleTime = runtimeGroup.MaximumCycleTime;  
runtimeGroup.MaximumCycleTime= 120000;
```

Verwendung

Gibt die maximale Zykluszeit der Runtime-Gruppe zurück und legt diese fest. Liegt der festgelegte Wert nicht im gültigen Bereich, wird eine Ausnahme erzeugt.

F-Ablaufgruppeninfo-DB

Programmcode

```
string infoDbName = runtimeGroup.InfoDbName;  
runtimeGroup.InfoDbName = "RTGSysInfo";
```

Verwendung

Gibt den Namen des bereits bestehenden Informations-DB (RTGSysInfoDB) der Ablaufgruppe zurück bzw. legt ihn fest. Mit dieser Methode wird kein neuer DB erstellt. Wenn der gesetzte Name des Informations-DB ungültig ist, wird eine Ausnahme erzeugt.

Vorverarbeitung

Programmcode

```
string preProcessingName = runtimeGroup.PreProcessingName;  
runtimeGroup.PreProcessingName = "PreProcessing";
```

Verwendung

Gibt den Namen des Vorverarbeitungs-FC zurück bzw. legt ihn fest. Wenn der Vorverarbeitungs-FC nicht festgelegt ist, wird der Wert null zurückgegeben. Wenn der gesetzte Name der Vorverarbeitung ungültig ist, wird eine Ausnahme erzeugt. Wenn der Name leer oder auf null gesetzt ist, wird die Vorverarbeitung zurückgesetzt, aber nicht der referenzierte FC gelöscht.

Nachverarbeitung

Programmcode

```
string postProcessingName = runtimeGroup.PostProcessingName;  
runtimeGroup.PostProcessingName = "PostProcessing";
```

Verwendung

Gibt den Namen des Nachverarbeitungs-FC zurück bzw. legt ihn fest. Wenn der Nachverarbeitungs-FC nicht festgelegt ist, wird der Wert null zurückgegeben. Wenn der gesetzte Name der Nachverarbeitung ungültig ist, wird eine Ausnahme erzeugt. Wenn der Name leer oder auf null gesetzt ist, wird die Nachverarbeitung zurückgesetzt, aber nicht der referenzierte FC gelöscht.

Globalen F-Peripherie-Statusbaustein erstellen

Programmcode

```
PlcBlock block = runtimegroup.GenerateGlobalFIOStatusBlock();
```

Verwendung

Erstellt einen globalen F-Peripherie-Statusbaustein oder überschreibt ihn, wenn der Statusbaustein bereits besteht.

Siehe auch: Handbuch "SIMATIC Safety - Projektieren und Programmieren (<https://support.industry.siemens.com/cs/ww/de/view/54110126>)" im Kapitel "Globalen F-Peripherie-Statusbaustein anlegen".

5.27.5.9 Sicherheitseinstellungen

Sicherheitseinstellungen

Programmcode

```
SafetySettings = safetyAdministration.Settings;
```

Verwendung

Die Eigenschaft `SafetySettings` gibt einen Container für die folgenden Eigenschaften zurück, auf die von der zurückgegebenen Instanz zugegriffen werden kann.

Zuweisung von Bausteinnummern, die vom Sicherheitssystem generiert werden

Programmcode

```
AssignmentOfBlockNumbers blockNumbers =  
settings.AssignmentOfBlockNumbers;  
ushort fromFBNumber = blockNumbers.FromFB;  
ushort toFBNumber = blockNumbers.ToFB;  
ushort fromFCNumber = blockNumbers.FromFC;  
ushort toFCNumber = blockNumbers.ToFC;  
ushort fromDBNumber = blockNumbers.FromDB;  
ushort toDBNumber = blockNumbers.ToDB;
```



```
BlockNumbersManagementMode mode = blockNumbers.ManagementMode;  
if (mode != BlockNumbersManagementMode.FSystemManaged) {  
    // Set block numbers...  
}
```

Verwendung

Ruft die vom Sicherheitssystem erzeugten Bausteinnummern ab oder legt diese fest. Wird eine Nummer festgelegt und der Wert liegt nicht im gültigen Bereich, wird eine Ausnahme erzeugt. Wird eine Nummer festgelegt und der Managementmodus ist `FSystemManaged`, wird eine Ausnahme erzeugt.

Safety-System-Version

Programmcode

```
SafetySystemVersion version = settings.SafetySystemVersion;  
IList<SafetySystemVersion> versions =  
settings.GetApplicableSafetySystemVersions();  
  
settings.SafetySystemVersion = versions.Last();  
  
SafetySystemVersion version1_6 = versions.SingleOrDefault(v =>  
v.Value == "V1.6");  
if (version1_6 != null) {  
    settings.SafetySystemVersion = version1_6;  
}
```

Verwendung

Ruft die Safety-System-Version der F-CPU ab bzw. legt diese fest.

Mit `GetApplicableSafetySystemVersions()` kann eine Liste der gültigen Safety-System-Versionen für die entsprechende F-CPU abgerufen werden. Die empfangene Liste ist aufsteigend nach Versionsnummer geordnet. Mit `.Last()` wird die höchste verfügbare Versionsnummer abgerufen.

Aus der Liste der gültigen Safety-System-Versionen kann auch eine bestimmte Version ausgewählt werden.

Der Sicherheitsbetrieb kann deaktiviert werden

Programmcode

```
bool safetyModeCanBeDisabled = settings.SafetyModeCanBeDisabled;  
settings.SafetyModeCanBeDisabled = true;
```

Verwendung

Ruft ab oder legt fest, ob der Safety-Modus deaktiviert werden kann.

Aktivierung der F-Änderungshistorie**Programmcode**

```
bool activationOfFChangeHistory =  
settings.ActivationOfFChangeHistory;  
settings.ActivationOfFChangeHistory = true;
```

Verwendung

Ruft ab oder legt fest, ob die F-Änderungshistorie aktiviert ist.

Konsistentes Hochladen aktivieren**Programmcode**

```
bool consistentUpload = (bool)  
settings.GetAttribute("EnableConsistentUploadFromFCpu");  
settings.SetAttribute("EnableConsistentUploadFromFCpu", true);
```

Verwendung

Ruft ab oder legt fest, ob das konsistente Hochladen von einer F-CPU aktiviert ist. Diese Einstellung gilt nur für das Lesen und Schreiben in bestimmten F-CPU's.

ID-Variable für F-Kommunikation aktivieren**Programmcode**

```
bool fCommunicationIdTag = (bool)  
settings.GetAttribute("EnableFCommunicationIdTag");  
settings.SetAttribute("EnableFCommunicationIdTag", true);
```

Verwendung

Ruft ab oder legt fest, ob die ID-Variable der F-Kommunikation aktiviert ist.

Für die Sicherheitsbewertung dieser Information ist Handbuch "SIMATIC Safety – Projektierung und Programmierung" zu beachten.

Treiber-Instanzdatenbausteine ohne Präfix anlegen

Programmcode

```
bool prefix = settings.CreateDriverInstanceDataBlocksWithoutPrefix;  
settings.CreateDriverInstanceDataBlocksWithoutPrefix = true;
```

Verwendung

Ruft ab oder legt fest, ob die Namen der F-Peripherie-Instanz-DBs ohne Präfix angelegt werden.

Bereinigen vom System generierter Objekte

Programmcode

```
settings.CleanSystemGeneratedObjects();
```

Verwendung

Bereinigt die vom System beim Übersetzen generierten Objekte.

5.27.6 SafetyPrintout

Diese Funktion ist für S7-1200/1500 F-CPU's verfügbar.

Programmcode

```
Den Dienst SafetyPrintout können Sie von einer verfügbaren DeviceItem-Instanz  
beziehen. Anschließend rufen Sie die Funktion Print() für den Dienst auf. Bei erfolgreicher  
Ausführung gibt die Funktion "true" zurück.  
SafetyPrintout printoutService =  
deviceItem.GetService<SafetyPrintout>();  
if (printoutService != null) {  
    FileInfo path = new FileInfo(@"C:\safety_printout.pdf");  
    bool success =  
printoutService.Print(SafetyPrintoutFilePrinter.MicrosoftPrintToPdf,  
    path, "DocuInfo_ISO_A4_Portrait", SafetyPrintoutOption.All)  
}
```

Anwendung

Mit der Aktion `Print(...)` des Dienstes `SafetyPrintout` wird der Sicherheitsausdruck mit den angegebenen Parametern ausgelöst. Bei erfolgreichem Ausdruck gibt die angegebene Funktion "true" zurück.

Der angegebene Pfad muss ein gültiger Pfad zum Speichern des Ausdrucks sein.

Für den Ausdruck kann ein bestimmtes `documentLayout` (z. B. "DocuInfo_ISO_A4_Portrait") festgelegt werden. Außerdem kann über den Parameter `documentationOption` ein Sicherheitsausdruck mit den Optionen "All" oder "Compact" ausgewählt werden.

Außerdem wird der ausgewählte digitale Drucker (Enum `SafetyPrintoutFilePrinter`) im System zum Erstellen des digitalen Sicherheitsausdrucks verwendet. Hier können die Drucker "Microsoft Print to PDF" oder "Microsoft XPS Document Writer" ausgewählt werden. Diese Drucker sind unter Windows 10 verfügbar und standardmäßig aktiviert ("Microsoft XPS Document Writer" seit Windows 7). Damit der Ausdruck erfolgreich ist, muss der ausgewählte Drucker aktiviert werden. Ist dies nicht der Fall, können diese Drucker im Fenster für die Windows-Funktionen aktiviert werden, indem erst die Tastenkombination *Windows-Taste + R* gedrückt und dann folgende Datei ausgeführt wird:
`optionalfeatures.exe`

Export/Import

6.1 Überblick

6.1.1 Grundlagen zum Import/Export

Einleitung

Sie können bestimmte Konfigurationsdaten exportieren und die Daten nach dem Editieren in dasselbe oder ein anderes Projekt reimportieren.

Hinweis

Im Zusammenhang mit dieser Beschreibung gibt es keinerlei Verpflichtungen oder Garantien, die Quelldatei manuell zu ändern und auszuwerten. Siemens übernimmt deshalb keine Haftung für den Gebrauch dieser Beschreibung als Ganzes oder in Teilen.

Exportierbare und importierbare Objekte

Die folgenden Konfigurationsdaten können auch mit TIA Portal Openness APIs importiert oder exportiert werden:

Tabelle 6-1 Projekte

Objekte	Export	Import
Grafiksammlung	X	X

Tabelle 6-2 PLC

Objekte	Export	Import
Bausteine	X	X
Knowhow-geschützte Bausteine	X	–
Fehlersichere Bausteine	X	X
Systembausteine	X	–
PLC-Variablentabellen	X	X
Technologieobjekte	X	X
PLC-Variablen und -Konstanten	X	X
Anwenderdatentypen	X	X
DBs	X	-

Tabelle 6-3 HMI

Objekte	Export	Import
Bilder	X	X
Bildvorlagen	X	X
Globale Bilder	X	X
Pop-up-Bilder	X	X
Slide-in-Bilder	X	X
Skripte	X	X
Textlisten	X	X
Grafiklisten	X	X
Zyklen	X	X
Verbindungen	X	X
Variablentabelle	X	X
Variablen	X	X

Vollständiger Export oder Export offener Referenzen

Die oben aufgeführten Objekttypen werden mit allen Objekten exportiert oder importiert, wenn diese demselben Unterbaum angehören. Diese Regel gilt auch für referenzierte Objekte im selben Unterbaum.

Bei referenzierten Objekten in anderen Unterbäumen ist jedoch ein vollständiger Export oder Import nicht möglich. Stattdessen werden "offene Referenzen" zu diesen Objekten exportiert oder importiert.

Referenzierte Objekte aus demselben Unterbaum werden nur exportiert, wenn sie der Gruppe exportierbarer Objekte angehören. Alle Dynamisierungen an Objekten werden beim Importieren/Exportieren als Objekte behandelt und ebenfalls exportiert/importiert.

Der Export umfasst alle Objektattribute, die bei der Konfiguration geändert wurden. Das gilt unabhängig davon, ob das geänderte Attribut genutzt wird oder nicht.

Beispiel: Sie haben ein grafisches EA-Feld mit dem Modus "Eingabe/Ausgabe" konfiguriert und die Einstellung "Sichtbar nach Anklicken" für das Attribut "Rollbalken" ausgewählt. Im Laufe der Konfiguration haben Sie den Modus in "Zwei Zustände" geändert. Das Attribut "Rollbalken" ist in diesem Modus nicht verfügbar. Weil das Attribut "Rollbalken" geändert wurde, wird es in den Export einbezogen, selbst wenn das Attribut nicht genutzt wird.

Geöffnete Referenzen importieren

Sie können auch Objekte mit geöffneten Referenzen importieren (siehe Import von Projektierungsdaten (Seite 1255)).

Wenn die referenzierten Objekte im Zielprojekt enthalten sind, werden die geöffneten Referenzen automatisch wieder mit den Objekttypen verknüpft. Diese Objekte müssen am gleichen Ort verfügbar sein und dem gleichen Namen wie für den Export zugeordnet sein. Wenn die referenzierten Objekte nicht im Zielprojekt enthalten sind, können die geöffneten

Referenzen nicht aufgelöst werden. Es wird kein zusätzliches Objekt erzeugt, um diese geöffneten Referenzen aufzulösen.

Dateiformat exportieren und importieren

Das Dateiformat zum Exportieren und Importieren ist XML. Nur CAx-Daten erfordern das AML-Format. Die unterschiedlichen Schemadefinitionen für sämtliche Formate werden in dem jeweiligen Abschnitt in diesem Handbuch beschrieben:

- XML-Format für die Daten eines HMI-Geräts (Seite 1263)
- XML-Format für die Daten eines PLC-Geräts (Seite 1338)
- AML-Format für CAx-Daten (Seite 1478)

Schriftarten importieren und exportieren

Für Objekte definierte Schriftarten werden ebenfalls exportiert und importiert.

Wenn Sie nicht im Projekt enthaltene Schriftarten importieren, wird nach dem Import die Standardschriftart am Objekt angezeigt. Jedoch wird die importierte Schriftart in der Datenverwaltung gespeichert.

Wenn die Attribute für eine Schriftart nicht in einer Importdatei zuwiesen sind, werden die Attribute nach dem Import mit Standardwerten belegt.

Technologieobjekte importieren und exportieren

Welche Technologieobjekte ab TIA Portal Version V16 exportiert und importiert werden können, finden Sie in Kapitel "Übersicht über Technologieobjekte und Versionen".

Einschränkungen

Das Exportformat ist intern und ausschließlich für die aktuelle Version von TIA Portal Openness gültig. Das Exportformat kann sich in künftigen Versionen ändern.

Alle beim Import und Export auftretenden Fehler werden als Ausnahmen gemeldet. Weitere Informationen über Ausnahmen siehe Abschnitt Umgang mit Exceptions (Seite 1229).

Siehe auch

Einsatzgebiet von Import/Export (Seite 1252)

Export von Projektierungsdaten (Seite 1253)

6.1.2 Einsatzgebiet von Import/Export

Einleitung

Die Funktionalität zum Importieren/Exportieren ermöglicht Ihnen, bestimmte Objekte gezielt zu exportieren.

Sie können die exportierten Daten in einem externen Programm editieren oder unverändert in anderen TIA-Portal-Projekten wiederverwenden.

Wenn Sie die Importdatei richtig strukturieren, können Sie auch extern erzeugte Konfigurationsdaten importieren, ohne zuerst einen Export durchführen zu müssen.

Hinweis

Wenn Sie extern erzeugte Konfigurationsdaten mit Codefehlern oder falscher Struktur importieren, kann dies unerwartete Fehler verursachen.

Anwendungsbereich

Exportieren und Importieren von Daten ist nützlich für folgende Aufgaben:

- Externes Editieren von Konfigurationsdaten.
- Importieren von extern erzeugten Konfigurationsdaten, z. B. Textlisten und Variablen.
- Verteilen von spezifizierten Konfigurationsdaten an verschiedene Projekte, z. B. ein geändertes Prozessbild, das in mehreren Projekten verwendet werden soll.
- Zum Replizieren und Anpassen der Hardware-Konfiguration zwischen dem TIA-Portal-Projekt und einem ECAD-Programm.

Siehe auch

Grundlagen zum Import/Export (Seite 1249)

6.1.3 Versionsspezifischer SimaticML-Import

Einleitung

Die einzelnen Versionen der Openness API unterstützen den Export von SIMATIC-ML-Dateien. Die Version der exportierten SIMATIC-ML-Datei sollte mit der Version des TIA Portals und nicht mit der Version der Openness API übereinstimmen.

Die SimaticML-Dateiversion für den Export entspricht immer der verwendeten TIA Portal-Version, unabhängig davon, welche Openness API-Version verwendet wird. Bis TIA Portal V18 ist der Import von SimaticML-Dateien nur möglich, wenn die verwendete Openness API-Version höher oder gleich der SimaticML-Version ist.

Seit TIA Portal V19 können alle SimaticML-Versionen mit allen Openness API-Versionen

importiert werden, solange die SimaticML-Version innerhalb des LTS-Gültigkeitsbereichs liegt (d. h. ab Vxx-3 bis Vxx). Dies ermöglicht langfristig stabile Umläufe.

Hinweis

Die Versionsnummer Vxx bezieht sich auf die aktuell installierte Version von TIA Portal.

6.1.4 Bearbeiten der XML-Datei

Einleitung

Zur Bearbeitung einer XML-Datei für den Import von Projektierungsdaten verwenden Sie einen XML-Editor oder einen Texteditor.

Wenn Sie umfangreiche Änderungen durchführen oder selbst Objektstrukturen aufbauen, empfehlen wir die Verwendung eines XML-Editors mit Auto-Vervollständigungsfunktion.

Hinweis

Ändern des XML-Inhalts setzt umfangreiche Kenntnisse der Struktur und der Validierungsregeln in XML voraus. Vermeiden Sie Validierungsfehler und arbeiten Sie nur in Ausnahmefällen manuell in der XML-Struktur.

6.1.5 Export von Projektierungsdaten

Einleitung

Die Konfigurationsdaten jedes Startobjekts (Stamm) werden separat in eine XML-Datei exportiert.

Zum Editieren der Exportdatei sind angemessene Kenntnisse von XML erforderlich. Verwenden Sie für komfortables Editieren einen XML-Editor.

Beispiel

Sie haben ein Prozessbild, das ein EA-Feld enthält. In dem EA-Feld ist eine externe Variable konfiguriert. Ein Export des Prozessbilds enthält das Bild und das EA-Feld. Die Variable und die von der Variable verwendete Verbindung werden nicht exportiert. Stattdessen wird nur eine offene Referenz in den Export einbezogen.

Inhalt der Exportdatei

Beginnend mit dem Startobjekt werden alle Objekte eines Unterbaums und deren Attribute in der Exportdatei gespeichert. Alle Verweise auf Objekte anderer Unterbäume werden nur als offene Verweise exportiert. Die entsprechenden Attribute der referenzierten Objekte in verschiedenen Unterbäumen werden nicht in die Exportdatei geschrieben.

Hinweis

Der Export von Objekttypen aus der Bibliothek wird nicht unterstützt.

Sie können Objekte als Typ in der Bibliothek erzeugen. Instanzen des im Projekt verwendeten Objekttyps können wie andere Objekte mit der TIA Portal Openness-Anwendung bearbeitet werden. Wenn Sie Objekte exportieren, werden die Instanzen ohne Typinformationen exportiert.

Wenn Sie diese Objekte wieder in das Projekt importieren, werden die Instanzen des Objekttyps überschrieben und die Instanz wird vom Objekttyp abgelöst.

Die Exportdatei enthält nicht notwendigerweise alle Attribute eines Objekts. Sie legen fest, welche Daten exportiert werden sollen:

- `ExportOptions.None`
Diese Einstellung exportiert nur die geänderten Daten oder die Daten, die von den Standardwerten abweichen.
Die Exportdatei enthält auch alle Werte, die für den anschließenden Datenimport obligatorisch sind.
- `ExportOptions.WithDefaults`¹
Auch die Standardwerte werden exportiert.
- `ExportOptions.WithReadOnly`¹
Auch die schreibgeschützten Werte werden exportiert.

¹: Sie können diese beiden Optionen mit der folgenden

Syntax kombinieren: `Export (path, ExportOptions.WithDefaults | ExportOptions.WithReadOnly) ;`

Sämtliche Inhalte der Exportdatei liegen auf Englisch vor. Unabhängig davon werden enthaltene Projektttexte in alle vorhandenen Sprachen exportiert und importiert.

Alle Konfigurationsdaten werden als XML-Objekte in der Exportdatei modelliert.

Siehe auch

Grundlagen zum Import/Export (Seite 1249)

Bausteine exportieren (Seite 1395)

6.1.6 Import von Projektierungsdaten

Einleitung

Konfigurationsdaten werden aus einer zuvor exportierten und editierten XML-Datei oder aus einer von Ihnen selbst erzeugten XML-Datei importiert. Die in dieser Datei enthaltenen Daten werden beim Import geprüft. Dieser Ansatz verhindert, dass die Konfigurationsdaten im TIA Portal infolge des Imports inkonsistent werden.

Einschränkungen

- Alle Stammobjekte in der Importdatei müssen vom gleichen Typ sein, z. B. Variablentabellen und Bausteine.
- Wenn eine Importdatei mehrere Stammobjekte enthält und eines davon nicht gültig ist, werden sämtliche Inhalte der Importdatei nicht importiert.
- Beim Importieren von Texten müssen die entsprechenden Projektsprachen im Zielprojekt eingerichtet worden sein, um ein Fehlschlagen des Imports zu vermeiden. Bei Bedarf können Sie die Spracheinstellungen über TIA Portal Openness ändern oder beim Importieren neue kulturspezifische Enum-Optionen verwenden, um ein Fehlschlagen des Imports zu verhindern.
- Die kulturspezifischen Enum-Optionen `ImportOptions.ActivateInactiveCultures` und `ImportOptions.SkipInactiveCultures` werden bei Excel-basierten Import-APIs nicht unterstützt (Beispiel: `ImportSupervisionSettingsFromXlsx`). Die Verwendung dieser Optionen führt zum Fehlschlagen des Imports.
- Der Import schlägt beim Importieren von Texten fehl, wenn die entsprechenden Sprachen in TIA Portal nicht unterstützt werden.
- Wenn Sie ungültige Attribute eines Objekts in der Importdatei angeben, die nicht in der grafischen Benutzeroberfläche des TIA Portals editiert werden können, wird der Import abgebrochen.
- Nur die im Feld "separately for each connection" aufgeführten Bereichszeiger können importiert oder exportiert werden.
- Der Import von Objekttypen aus der Bibliothek wird nicht unterstützt. Sie können Objekte als Typ in der Bibliothek erzeugen. Instanzen des im Projekt verwendeten Objekttyps können wie andere Objekte mit der TIA Portal Openness-Anwendung bearbeitet werden. Wenn Sie Objekte exportieren, werden die Instanzen ohne Typinformationen exportiert. Wenn Sie diese Objekte wieder in das Projekt importieren, werden die Instanzen des Objekttyps überschrieben und die Instanz wird vom Objekttyp abgelöst.

Hinweis

Geräteabhängige Wertebereiche für grafische Attribute

Wenn Werte grafischer Attribute außerhalb des gültigen Wertebereichs liegen, werden diese Werte beim Import auf die möglichen Maximalwerte für das HMI-Gerät zurückgesetzt.

Unterschiedliches Importverhalten

Wenn bereits im Projekt vorhandene Objekte importiert werden sollen, müssen Sie das Importverhalten mit verschiedenen Programmcodes steuern. Andernfalls werden die Objekte beim Import wieder im Projekt erzeugt.

Die folgenden Einstellungen für das Importverhalten sind möglich:

- `ImportOptions.None`
Bei dieser Einstellung werden die Konfigurationsdaten ohne Überschreiben importiert. Wenn ein Objekt aus einer bereits im Projekt vorhandenen XML-Datei importiert wird, wird der Import unterbrochen und eine Ausnahme ausgelöst.
- `ImportOptions.Override`
Bei Verwendung dieser Einstellung werden die Konfigurationsdaten mit automatischem Überschreiben importiert. Sie können angeben, dass im Projekt vorhandene Objekte beim Import überschrieben werden sollen. Relevante Objekte werden vor dem Import gelöscht und mit Standardwerten neu erzeugt. Diese Standardwerte werden beim Import mit den importierten Werten überschrieben. Wenn das vorhandene Objekt und das neue Objekt nicht in derselben Gruppe sind, kann kein Überschreiben stattfinden. Um Namenskonflikte zu vermeiden, wird der Import abgebrochen und eine Ausnahme ausgelöst.
- `ImportOptions.SkipInactiveCultures`
Mit dieser Einstellung werden die Konfigurationsdaten importiert, indem derjenige Text übersprungen wird, für den die erforderlichen Projektsprachen im Projekt nicht aktiv sind.
- `ImportOptions.ActivateInactiveCultures`
Mit dieser Einstellung werden die Konfigurationsdaten importiert, wobei automatisch die erforderlichen Projektsprachen aktiviert werden, die dem zu importierenden Text entsprechen, sofern sie nicht bereits im Projekt aktiviert sind.

Hinweis

Wenn Sie mehrere Importverhaltensweisen kombinieren möchten, können Sie Kombinationen von Optionen angeben (mit einem bitweisen OR-Operator). Sie können jedoch nicht die Optionen `ActivateInactiveCultures` und `SkipInactiveCultures` kombinieren, die einander widersprechen. Deren Kombination führt zu einem Fehlschlagen des Imports. Die Verwendung von bitweisem AND in der Kombination führt unabhängig von der gewählten Kombination immer zu `ImportOptions.None`.

Vorgehensweise zum Importieren

Wenn Sie eine XML-Datei importieren möchten, müssen die darin enthaltenen Daten bestimmten Regeln entsprechen. Die Inhalte der Importdatei müssen wohlgeformt sein. Es dürfen keine Syntaxfehler und keine Datenstrukturfehler vorhanden sein. Verwenden Sie bei umfangreichen Änderungen einen XML-Editor, der diese Kriterien vor dem Import überprüft.

Beim Import der XML-Datei in das TIA Portal werden die Daten in der Datei zuerst auf formale Fehler im XML-Code geprüft. Wenn bei dieser Prüfung Fehler erkannt werden, wird der Import abgebrochen und die Fehler werden in einer Ausnahme angezeigt (siehe Umgang mit Exceptions (Seite 1229)).

Siehe auch

Grundlagen zum Import/Export (Seite 1249)

Anwenderdatentyp importieren (Seite 1434)

6.2 Import/Export von Projektdaten

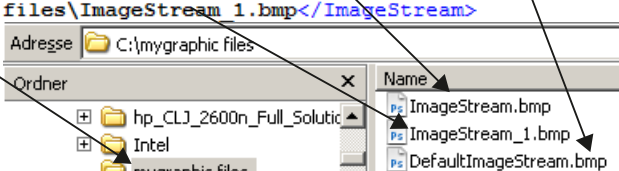
6.2.1 Grafiksammlung

6.2.1.1 Export/Import von Grafiken

Einleitung

Der Export von Konfigurationsdaten aus dem TIA Portal in die XML-Datei beinhaltet keine ausgewählten Grafiken oder Grafiken, auf die durch ein Objekt verwiesen wird. Die Grafiken werden beim Export separat gespeichert. In der XML-Datei wird anhand eines relativen Pfads und die Dateinamen auf die Grafiken verwiesen. Eine Grafikreferenz wird in der XML-Datei als ein Objekt modelliert und enthält eine Attributliste und bei Bedarf eine Verknüpfungsliste wie bei den anderen Objekten.

```
<Hmi.Globalization.MultiLingualGraphic ID="0">
  <AttributeList>
    <DefaultDithering>False</DefaultDithering>
    <DefaultImageStream external="path">mygraphic files\DefaultImageStream.bmp</DefaultImageStream>
    <DefaultSmoothness>False</DefaultSmoothness>
    <Name>MyGraphic1</Name>
  </AttributeList>
  <ObjectList>
    <Hmi.Globalization.GraphicItem ID="1" CompositionName="Items">
      <AttributeList>
        <Culture>en-US</Culture>
        <Dithering>False</Dithering>
        <ImageStream external="path">mygraphic files\ImageStream.bmp</ImageStream>
        <Smoothness>False</Smoothness>
      </AttributeList>
    </Hmi.Globalization.GraphicItem>
    <Hmi.Globalization.GraphicItem ID="2" CompositionName="Items">
      <AttributeList>
        <Culture>de-DE</Culture>
        <Dithering>False</Dithering>
        <ImageStream external="path">mygraphic files\ImageStream_1.bmp</ImageStream>
        <Smoothness>False</Smoothness>
      </AttributeList>
    </Hmi.Globalization.GraphicItem>
  </ObjectList>
</Hmi.Globalization.MultiLingualGraphic>
```



Grafiken exportieren

Der Export von Konfigurationsdaten beinhaltet ausschließlich Grafiken, die für den Export direkt ausgewählt wurden. Die exportierbaren Grafiken werden für die jeweilige Sprache im TIA Portal gespeichert. Wird ein Projekt in mehreren Sprachen konfiguriert, dann werden alle verwendeten Sprachversionen exportiert.

Beim Grafikexport wird ein neuer Ordner im Exportdateiordner erstellt. Der Dateiordnername wird durch Verknüpfen des XML-Dateinamens mit "Dateien" aufgebaut. Dieser Ordner enthält die exportierten Grafiken. Falls dieser Ordner bereits vorhanden ist, wird ein neuer Ordner erstellt und durch eine laufende Nummer ergänzt.

Die Grafiken werden in demselben Dateiformat wie im Projekt gespeichert. Das Datenformat wird nicht geändert oder konvertiert und die Auflösung und Farbtiefe bleiben unverändert.

Die Kennung "default" wird als Dateierweiterung für die als Standardsprache ausgewählte Sprache verwendet.

Wenn der Ordner bereits eine Datei mit demselben Namen enthält, wird der Dateiname der exportierten Grafik durch eine laufende Nummer ergänzt.

Grafiken importieren

Beim Importieren von Grafiken gelten die folgenden Voraussetzungen:

- Die Grafiken müssen ein vom TIA Portal unterstütztes Dateiformat besitzen.
- Auf die Grafiken muss in der XML-Datei durch Angabe eines relativen Pfads verwiesen werden.

Nach dem Export einer Grafik kann diese mithilfe eines Grafikprogramms außerhalb des TIA Portals bearbeitet und anschließend wieder importiert werden.

Siehe auch

Grundlagen zum Import/Export (Seite 1249)

6.2.1.2 Grafiken eines Projektes exportieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können entweder eine einzelne Grafik oder alle Grafiken der Grafiksammlung eines Projekts in allen Sprachen exportieren. Während des Exports wird eine XML-Datei mit allen betroffenen Projektgrafikeinträgen erstellt und zusammen mit den exportierten Grafiken referenziert. Die entsprechenden Grafiken werden zusammen mit der XML-Datei im selben Verzeichnis des Dateisystems gespeichert.

Damit die exportierten Grafiken (*.jpg, *.bmp, *.png, *.ico, etc.) geändert werden können, sind diese Grafiken nicht schreibgeschützt.

Programmcode: Grafik exportieren

Um die erforderliche Grafik zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all language variants of a single graphic
Project project = ...;
MultiLingualGraphicComposition graphicsComposition = project.Graphics;
MultiLingualGraphic graphic = graphicsComposition.Find("graphicName");
graphic.Export(new FileInfo(@"D:\ExportFolder\graphicName.xml"),
ExportOptions.WithDefaults);
```

Programmcode: Alle Grafiken exportieren

Um alle Grafiken einer Grafiksammlung zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all graphics of a graphic library
Project project = ...;
MultiLingualGraphicComposition graphicsComposition = project.Graphics;
foreach(MultiLingualGraphic graphic in graphicsComposition)
{
    graphic.Export(new FileInfo(string.Format(@"D:\Graphics\{0}.xml", graphic.Name)),
ExportOptions.WithDefaults);
}
```

6.2.1.3 Grafiken in ein Projekt importieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Eine XML-Datei wird zusammen mit den Sprachversionen einer Grafik in einem Verzeichnis Ihres Dateisystems gespeichert.

Sie können alle Grafiken in einem relativen Pfad in der XML-Datei referenzieren.

Sie können jetzt alle Sprachversionen einer in der XML-Datei enthaltenen Grafik in die Grafiksammlung importieren.

Beachten Sie auch Folgendes: Import von Projektierungsdaten (Seite 1255).

Programmcode

Um eine oder mehrere Grafiken zu importieren, ändern Sie den folgenden Programmcode:

```
//Import all language variants of a single graphic
Project project = ...;
MultiLingualGraphicComposition graphicComposition = project.Graphics;
graphicComposition.Import(new FileInfo(@"D:\Graphics\Graphic1.xml"),
ImportOptions.Override);
```

6.2.2 Projekttexte

6.2.2.1 Export von Projekttexten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Im TIA Portal finden Sie Projekttexte unter dem Knoten "Sprachen & Ressourcen" eines Projekts. Diese Texte werden in eine *.xlsx-Datei exportiert, die zum Beispiel für Übersetzungen genutzt werden kann. Die Einschränkungen beim Exportieren und Importieren von Projekttexten sind die gleichen wie in der Benutzeroberfläche. Diese Einschränkungen umfassen:

- Exportierte Texte können nur in das Projekt importiert werden, aus dem sie exportiert wurde.
- Texte können nur in Sprachen übersetzt werden, die im Projekt verfügbar sind. Bei Bedarf können Sie Projektsprachen über TIA Portal Openness hinzufügen.
- Nur vorhandene Texte können reimportiert werden. Wenn Texte aus dem ursprünglichen Projekt gelöscht oder neu erzeugt wurden, schlägt der Import dieser Texte fehl.

Sie müssen die folgenden Parameter festlegen:

Name	Beispiel	Beschreibung
pah	new FileInfo("D:\Test\Project-Text.xlsx")	Pfad zur Exportdatei
sourceLanguage	new CultureInfo("en-US")	Referenzsprache, aus der Text zu übersetzen ist
targetLanguage	new CultureInfo("de-DE")	Zielsprache, in die Text zu übersetzen ist

Hinweis

Mehrsprachige Texte werden zusammen mit dem übergeordneten Objekt, zu dem sie gehören, exportiert. Mehrsprachige Texte können nicht explizit exportiert werden.

Programmcode: Aus dem Knoten "Sprachen & Ressourcen" exportieren

Die Verwendung der Beispielparameter führt zum folgenden Programmcode zum Exportieren von Projekttextrn:

```
project.ExportProjectTexts(new FileInfo(@"D:\Test\ProjectText.xlsx"), new CultureInfo("en-US"), new CultureInfo("de-DE"));
```

XML-Struktur eines exportierten mehrsprachigen Textelements

```
...
<MultilingualText ID="2" CompositionName="Comment">
  <ObjectList>
    <MultilingualTextItem ID="3" CompositionName="Items">
      <AttributeList>
        <Culture>en-US</Culture>
        <Text>My super tag</Text>
      </AttributeList>
    </MultilingualTextItem>
    <MultilingualTextItem ID="4" CompositionName="Items">
      <AttributeList>
        <Culture>ru-RU</Culture>
        <Text>Мой супер тэг</Text>
      </AttributeList>
    </MultilingualTextItem>
  </ObjectList>
</MultilingualText>
...
```

6.2.2.2 Import von Projekttextrn**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Im TIA Portal finden Sie Projektttexte unter dem Knoten "Sprachen & Ressourcen" eines Projekts. Sie können Projektttexte aus einer *.xlsx-Datei importieren, was zum Beispiel für Übersetzungen genutzt werden kann. Die Einschränkungen beim Exportieren und Importieren von Projektttexten sind die gleichen wie in der Benutzeroberfläche. Diese Einschränkungen umfassen:

- Exportierte Texte können nur in das Projekt importiert werden, aus dem sie exportiert wurde.
- Nachdem ein Projekt mit "Speichern unter..." unter einem neuen Namen gespeichert wurde, ist der Import von Texten nicht mehr möglich.
- Übersetzte Texte können nur in Sprachen importiert werden, die in dem Projekt, aus dem sie exportiert wurden, verfügbar sind.
- Nur vorhandene Texte können reimportiert werden. Wenn Texte aus dem ursprünglichen Projekt gelöscht oder neu erzeugt wurden, schlägt der Import dieser Texte fehl.

Sie müssen die folgenden Parameter festlegen:

Name	Beispiel	Beschreibung
path	new FileInfo(@"D:\Test\Project-Text.xlsx")	Pfad zur Importdatei
updateSourceLanguage	true	Wenn "true", wird der Text der Referenzsprache anhand der Exportdatei aktualisiert. Wenn "false", wird der Text der Referenzsprache nicht aktualisiert.

Hinweis

Mehrsprachige Texte werden zusammen mit dem übergeordneten Objekt, zu dem sie gehören, importiert. Mehrsprachige Texte können nicht explizit importiert werden.

Programmcode

Die Verwendung der Beispielparameter führt zum folgenden Programmcode zum Importieren von Projektttexten:

```
ProjectTextResult result = project.ImportProjectTexts(new  
FileInfo(@"D:\Test\ProjectText.xlsx"), true);
```

Der Import der Projektttexte gibt ein Objekt zurück, das den Status des Imports anzeigt und den Pfad angibt, in dem das Importprotokoll gespeichert wird. Auf diese Attribute kann mit dem folgenden Code zugegriffen werden:

```
ProjectTextResultState resultState = result.State;  
FileInfo logFilePath = result.Path;
```

6.3 Import/Export von Daten eines HMI-Geräts

6.3.1 Datenstruktur für Import/Export

6.3.1.1 Aufbau einer XML-Datei

Einleitung

Die Daten in der Exportdatei vom Import/Export sind mit Verweis auf eine Grundstruktur gegliedert.

Grundstruktur einer Exportdatei

Die Exportdatei wird in einem XML-Format erzeugt.

Die XML-Datei beginnt mit einer Dokumentinformation. Sie beinhaltet die Daten der rechner-spezifischen Installation, mit der das Projekt exportiert wurde.

Die Exportdatei ist in die folgenden zwei Abschnitte unterteilt:

- Informationen über das Dokument
In diesem Abschnitt können Sie Ihre eigenen Informationen über den Export in gültiger XML-Syntax eingeben. Der Inhalt wird beim Import ignoriert.
Zum Beispiel können Sie einen Baustein mit `<IntegrityInformation>...</IntegrityInformation>` einfügen und darin zusätzliche Informationen über die Validierung unterbringen. Nach Weiterleitung der XML-Datei kann der Empfänger anhand dieses Bausteins vor dem Import prüfen, ob die XML-Datei geändert wurde.
- Objekt
Dieser Abschnitt enthält die zu exportierenden Elemente.

Informationen über das Dokument

```

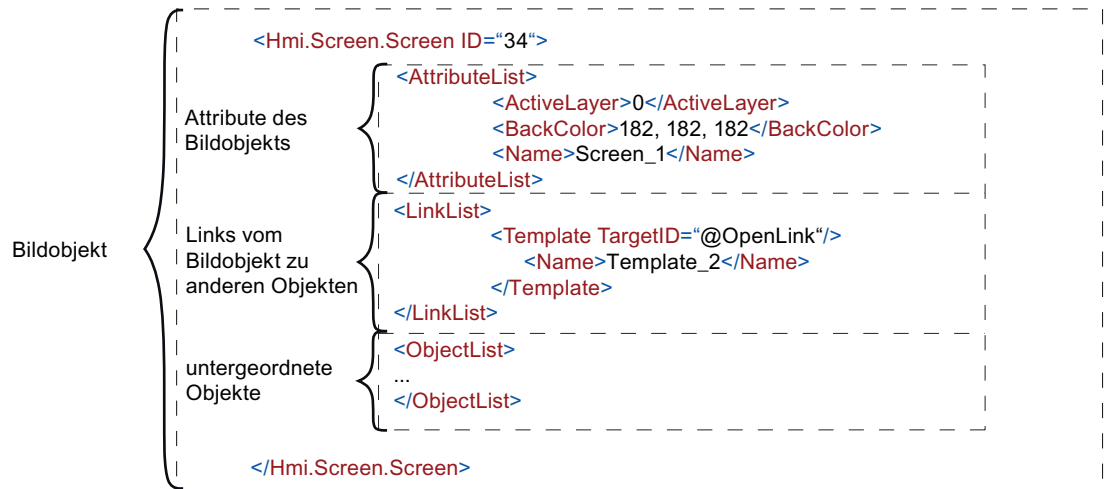
<?xml version="1.0" encoding="UTF-8" ?>
<Document xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <DocumentInfo>
    <UserName>Jane Doe</UserName>
    <Company>Example_Inc</Company>
    <IntegrityInformation>...</IntegrityInformation>
    <Created>2016-04-28T18:05:42.179207Z</Created>
    <ExportSetting>WithDefaults</ExportSetting>
    <InstalledProducts>
      <Product>
        <DisplayName>Totally Integrated Automation Portal</DisplayName>
        <DisplayVersion>V14</DisplayVersion>
      </Product>
      <OptionPackage>
        <DisplayName>WinCC Professional</DisplayName>
        <DisplayVersion>V14</DisplayVersion>
      </OptionPackage>
      <OptionPackage>
        <DisplayName>Siemens TIA Openness</DisplayName>
        <DisplayVersion>V14</DisplayVersion>
      </OptionPackage>
    </InstalledProducts>
  </DocumentInfo>
  <Hmi.Screen.Screen ID ="0">
    <AttributeList>
      <ActiveLayer>0</ActiveLayer>
      <BackColor>189,190,0</BackColor>
      <Height>422</Height>
      <Name>Root screen</Name>
      <Number>1</Number>
      <Visible>True</Visible>
      <Width>640</Width>
    </AttributeList>
    <LinkList>
      <Template TargetID="@OpenLink">
        <Name>Template_1</Name>
      </Template>
    </LinkList>
    <ObjectList>
      <Name>dummy</Name>
    </ObjectList>
  </Hmi.Screen.Screen>
</Document>

```

Bildobjekt

Bildobjekte einer Exportdatei

Die exportierten Elemente sind in zusätzlichen Elementen der XML-Datei verfügbar.



Siehe auch

Grundlagen zum Import/Export (Seite 1249)

6.3.1.2 Struktur der Daten für Import/Export

Objekte

Die Grundstruktur ist für alle Objekte gleich.

Jedes Objekt in der XML-Datei beginnt mit seinem Typ, zum Beispiel "Hmi.Screen.Button", und einer ID. Die ID wird beim Export automatisch erzeugt.

```
<Hmi.Screen.Button CompositionName="ScreenItems" ID="60">
```

Jedes Objekt mit Ausnahme des Startobjekts enthält auch ein XML-Attribut "CompositionName". Der Wert für dieses Attribut ist voreingestellt. Es ist gelegentlich notwendig, dieses Attribut anzugeben, zum Beispiel zum Ändern der Beschriftung beim Drücken oder Loslassen einer Schaltfläche.

```

<MultilingualText ID="A" CompositionName="TextOff">
  <ObjectList>
    <MultilingualTextItem ID="B" CompositionName="Items">
      <AttributeList>
        <Culture>en-US</Culture>
        <Text>
          <body>
            <p>TextOff</p>
          </body>
        </Text>
      </AttributeList>
    </MultilingualTextItem>
  </ObjectList>
</MultilingualText>
<MultilingualText ID="C" CompositionName="TextOn">
  <ObjectList>
    <MultilingualTextItem ID="D" CompositionName="Items">
      <AttributeList>
        <Culture>en-US</Culture>
        <Text>
          <body>
            <p>TextOn</p>
          </body>
        </Text>
      </AttributeList>
    </MultilingualTextItem>
  </ObjectList>
</MultilingualText>

```

Attribute

Jedes Objekt enthält Attribute, die in einem Abschnitt "AttributeList" enthalten sind. Jedes Attribut ist als XML-Element modelliert, z. B. "BackColor". Der Wert eines Attributs ist als XML-Inhalt modelliert, z. B. "204, 204, 204".

```

<Hmi.Screen.Button ID="2" CompositionName="ScreenItems">
  <AttributeList>
    <BackColor>204, 204, 204</BackColor>
    <ObjectName>Button_1</ObjectName>
  </AttributeList>
</Hmi.Screen.Button>

```

Bei referenzierenden Objekten enthält jedes Objekt gegebenenfalls einen Abschnitt "LinkList". Dieser Abschnitt enthält Verknüpfungen mit anderen Objekten innerhalb oder außerhalb der XML-Datei. Jede Verknüpfung ist als XML-Element modelliert. Die Bezeichnung einer Verknüpfung wird vom Zielobjekt in der Schemadatei festgelegt. Jede Verknüpfung enthält auch das Attribut "TargetID". Wenn das Zielobjekt in der XML-Datei enthalten ist, ist der Wert des Attributs "TargetID" die ID des referenzierten Objekts mit vorangestellter Raute "#". Wenn das Zielobjekt nicht in der XML-Datei enthalten ist, hat das Attribut "TargetID" den Wert "@OpenLink". Die tatsächliche Referenz des Objekts ist als untergeordnetes XML-Element modelliert.

```
<Hmi.Tag.Tag ID="17">
  <AttributeList>
    <Name>Tag_1</Name>
  </AttributeList>
  <LinkList>
    <AcquisitionCycle TargetID="@OpenLink">
      <Name>2 s</Name>
    </AcquisitionCycle>
    <Connection TargetID="@OpenLink">
      <Name>HMI_connection</Name>
    </Connection>
  </LinkList>
</Hmi.Tag.Tag>
```

Die nachstehenden Bilder zeigen die Beziehung zwischen der exportierten XML-Struktur und den zugeordneten Objekten in WinCC.

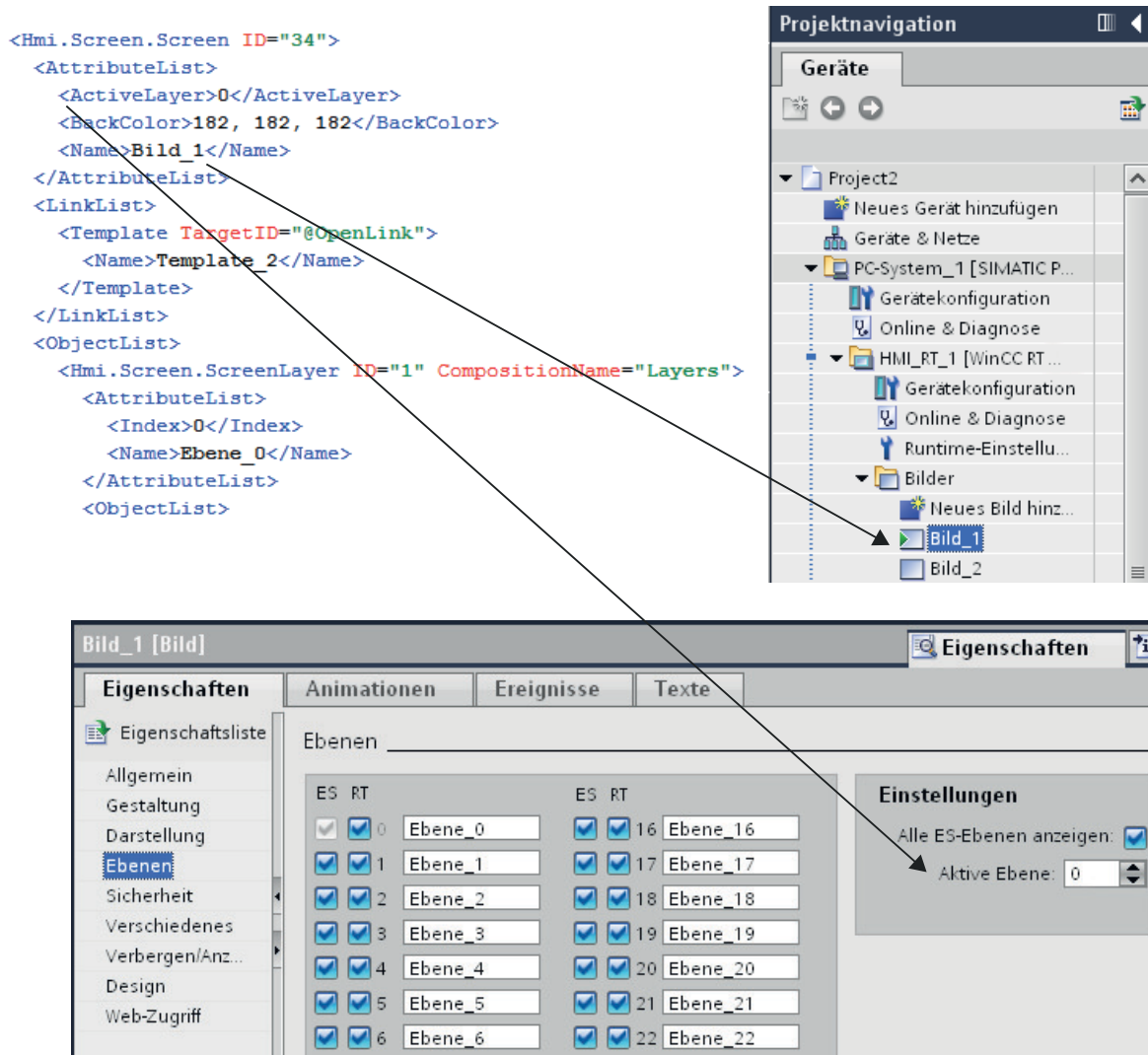


Bild 6-1 Beziehung zwischen WinCC-Benutzeroberfläche und XML-Struktur.

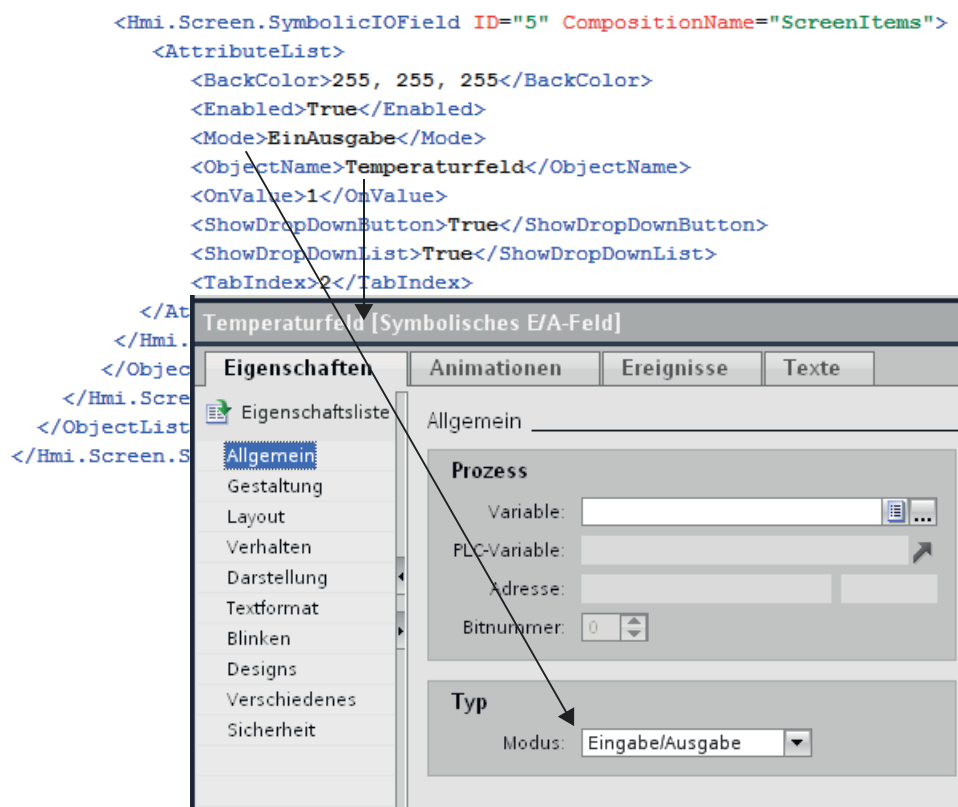


Bild 6-2 Beziehung zwischen den Einstellungen in WinCC und XML-Struktur.

6.3.1.3 Zyklen

Zyklen exportieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt den Export aller Zyklen eines bekannten HMI-Geräts in eine XML-Datei. Die Generierung der entsprechenden Exportdatei weist darauf hin, dass der Export abgeschlossen ist.

Programmcode

Um Zyklen aus einem HMI-Gerät in eine XML-Datei zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports cycles from an HMI device
private static void ExportCyclesFromHMITarget(HmiTarget hmitarget)
{
    CycleComposition cycles = hmitarget.Cycles;
    foreach(Cycle cycle in cycles)
    {
        cycle.Export(new FileInfo(string.Format(@"C:\Samples\{0}.xml", cycle.Name)),
ExportOptions.WithDefaults);
    }
}
```

Zyklen importieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Wenn Sie `ImportOptions.None` verwenden, können Sie anhand der Zusammensetzungszahl (Composition count) die Zyklen ermitteln, die tatsächlich importiert wurden. Sie haben Zugriff auf diese importierten Zyklen.

Hinweis

Standardzyklen haben Attribute, die in der Benutzeroberfläche nicht bearbeitet werden können. Wenn Sie in der Importdatei angeben, dass diese Attribute geändert werden sollen, verursacht der Importvorgang eine `NonRecoverableException` und schließt das TIA Portal.

Programmcode

Um einen Zyklus oder mehrere Zyklen aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports cycles to an HMI device
private static void ImportCyclesToHMITarget(HmiTarget hmitarget)
{
    CycleComposition cycles = hmitarget.Cycles;
    string dirPathImport = @"C:\OpennessSamples\Import\";
    string cycleImportFileName = "CycleImport.xml";
    string fullPath = Path.Combine(dirPathImport, cycleImportFileName);

    cycles.Import(new FileInfo(fullPath), ImportOptions.None);
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

6.3.2 Variablentabellen

6.3.2.1 HMI-Variablentabellen exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Pro HMI-Variablentabelle wird eine XML-Datei exportiert. Die API unterstützt diesen Exportvorgang. Der Export von Variablentabellen ist auch in Unterordnern verfügbar.

Programmcode: Alle HMI-Variablentabellen aus einem angegebenen Ordner exportieren

Um alle HMI-Variablentabellen aus einem bestimmten Ordner zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all tag tables from a tag folder
private static void ExportAllTagTablesFromTagFolder(HmiTarget hmitarget)
{
    TagSystemFolder folder = hmitarget.TagFolder;
    TagTableComposition tables = folder.TagTables;

    foreach (TagTable table in tables)
    {
        FileInfo info = new FileInfo(string.Format(@"C:\OpennessSamples\TagTables\{0}.xml",
table.Name));
        table.Export(info, ExportOptions.WithDefaults);
    }
}
```

Programmcode: Eine HMI-Variablentabelle exportieren

Um eine einzelne HMI-Variablentabelle zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports a tag table from an HMI device
private static void ExportTagTableFromHMITarget(HmiTarget hmitarget)
{
    string tableName = "Tag table XYZ";
    TagSystemFolder folder = hmitarget.TagFolder;
    TagTableComposition tables = folder.TagTables;
    TagTable table = tables.Find(tableName);

    if (table != null)
    {
        FileInfo info = new FileInfo(string.Format(@"C:\OpennessSamples\TagTables\{0}.xml",
table.Name));
        table.Export(info, ExportOptions.WithDefaults);
    }
}
```

Programmcode: Alle HMI-Variablentabellen exportieren

Um alle HMI-Variablentabellen zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all tag tables from an HMI device
private static void ExportAllTagTablesFromHMITarget(HmiTarget hmitarget)
{
    TagSystemFolder sysFolder = hmitarget.TagFolder;

    //First export the tables in underlying user folder
    foreach (TagUserFolder userFolder in sysFolder.Folders)
    {
        ExportUserFolderDeep(userFolder);
    }

    //then, export all tables in the system folder
    ExportTablesInSystemFolder(sysFolder);
}

private static void ExportUserFolderDeep(TagUserFolder rootUserFolder)
{
    foreach (TagUserFolder userFolder in rootUserFolder.Folders)
    {
        ExportUserFolderDeep(userFolder);
    }
    ExportTablesInUserFolder(rootUserFolder);
}

private static void ExportTablesInUserFolder(TagUserFolder folderToExport)
{
    TagTableComposition tables = folderToExport.TagTables;
    foreach (TagTable table in tables)
    {
        string fullPath = string.Format(@"C:\OpennessSamples\TagTables\{0}.xml",
table.Name);
        table.Export(new FileInfo(fullPath), ExportOptions.WithDefaults);
    }
}

private static void ExportTablesInSystemFolder(TagSystemFolder folderToExport)
{
    TagTableComposition tables = folderToExport.TagTables;
    foreach (TagTable table in tables)
    {
        string fullPath = string.Format(@"C:\OpennessSamples\TagTables\{0}.xml",
table.Name);
        table.Export(new FileInfo(fullPath), ExportOptions.WithDefaults);
    }
}
```

6.3.2.2 HMI-Variablentabelle importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Um die HMI-Variablentabelle einer XML-Datei in einen benutzerdefinierten Ordner oder in einen Systemordner zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports a single HMI tag table from a XML file
private static void ImportSingleHMITagTable(HmiTarget hmitarget)
{
    TagSystemFolder folder = hmitarget.TagFolder;
    TagTableComposition tables = folder.TagTables;

    FileInfo info = new FileInfo(@"D:\Samples\Import\myExportedTagTable.xml");
    tables.Import(info, ImportOptions.Override);
}
```

Fehlerhafter Import von Variablen

Wenn Sie in den Namen von Variablen oder referenzierten Variablen die folgenden Symbole verwenden, kommt es beim Import der Variablen zu Fehlern:

- . (Punkt)
- \ (Backslash)

Abhilfe 1:

Vergewissern Sie sich vor einem Export, dass der Name der zu exportierenden Variable oder Bezugsvariable keinen Punkt und keinen Backslash enthält.

Abhilfe 2:

Schließen Sie in der Importdatei die Namen von Variablen oder referenzierten Variablen mit Anführungszeichen aus.

Beispiel

- Variablenname mit Symbol:

```
<name>Siemens.Simatic.Hmi.Utah.Tag.HmiTag:41000_Options_Time_Date\DB_SFX0908_HMI1.Actual_Date_Time.Hour</name>
```

- Variablenname mit ausgeschlossenen Symbol in Anführungszeichen:

```
<name>"Siemens.Simatic.Hmi.Utah.Tag.HmiTag:41000_Options_Time_Date\DB_SFX0908_HMI1.Actual_Date_Time.Hour"</name>
```

6.3.2.3 Einzelne Variable einer HMI-Variablentabelle exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Die folgenden Objekttypen des Objektmodells können als untergeordnete Elemente einer HMI-Variable vorhanden sein und werden beim Export berücksichtigt:

MultilingualText	Für Kommentar, TagValue, DisplayName
TagArrayMemberTag	Für HMI-Array-Elemente
TagStructureMember	Für HMI-Strukturelemente
Event	Für konfigurierte Ereignisse
MultiplexEntry	Für konfigurierte Multiplex-Einträge von Variablen

Programmcode

Um eine einzelne Variable aus einer HMI-Variablentabelle in eine XML-Datei zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports a selected tag from a tag table
private static void ExportSelectedTagFromTagTable(HmiTarget hmitarget)
{
    TagSystemFolder tagFolder = hmitarget.TagFolder;
    TagTable mytable = tagFolder.TagTables.Find("MyTagTable");

    TagComposition containingTags = mytable.Tags;
    Tag myTag = containingTags.Find("MyTag");

    if (myTag != null)
    {
        FileInfo info = new FileInfo(string.Format(@"C:\OpennessSamples\Tags\{0}.xml",
myTag.Name));
        myTag.Export(info, ExportOptions.WithDefaults);
    }
}
```

6.3.2.4 Einzelne Variable in eine HMI-Variablentabelle importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Die folgenden Objektmodell-Objekttypen können als untergeordnete Elemente einer HMI-Variable vorhanden sein und werden beim Import berücksichtigt:

MultilingualText	Für Kommentar, TagValue, DisplayName
TagArrayMemberTag	Für HMI-Array-Elemente
TagStructureMember	Für HMI-Strukturelemente
Event	Für konfigurierte Ereignisse
MultiplexEntry	Für konfigurierte Multiplex-Einträge von Variablen

Programmcode

Um eine HMI-Variable aus einer XML-Datei in eine HMI-Variablentabelle zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports a tag into a tag table
private static void ImportTagIntoTagTable(HmiTarget hmitarget)
{
    TagSystemFolder tagFolder = hmitarget.TagFolder;
    TagTable myTable = tagFolder.DefaultTagTable;
    TagComposition tagComposition = myTable.Tags;

    FileInfo info = new FileInfo(@"D:\Samples\Import\myExportedTag.xml");
    tagComposition.Import(info, ImportOptions.Override);
}
```

6.3.2.5 Besonderheiten beim Export/Import von HMI-Variablen

Einleitung

Besondere Überlegungen gelten beim Export und Import der folgenden HMI-Variablen:

- Externe HMI-Variablen mit integrierter Verbindung
- HMI-Variablen mit dem Datentyp "UDT"

Ähnliche Programmcodes

Der Programmcode für die oben genannten HMI-Variablen ist fast identisch mit den folgenden Programmcodes:

- Programmcode: HMI-Variablen exportieren (Seite 1275)
- Programmcode: HMI-Variablen importieren (Seite 1276)

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Besondere Überlegungen für den Export/Import einer externen HMI-Variable mit integrierter Verbindung

Beim Exportieren einer externen HMI-Variable mit integrierter HMI-Verbindung wird statt der PLC-Variablendaten nur die Verknüpfung der HMI-Variable mit der PLC-Variable in der Exportdatei gespeichert.

Vor dem Import müssen Sie sicherstellen, dass der PLC, die entsprechenden PLC-Variablen und die integrierte Verbindung zum entsprechenden PLC im Projekt vorhanden sind. Wenn das nicht der Fall ist, müssen diese Elemente vor dem Import erzeugt werden. Beim anschließenden Import der externen HMI-Variable wird die Verknüpfung zur PLC-Variable wieder aktiviert.

Namen externer HMI-Variablen müssen über alle Variablen Tabellen eines Projekts eindeutig sein. Wenn Sie beim Import nicht die passende Variablen Tabelle für die HMI-Variable angeben, wird der Import abgebrochen.

Verwenden Sie zum Importieren einer externen HMI-Variable mit integrierter Verbindung die folgende XML-Struktur:

```
<Hmi.Tag.Tag ID="1" CompositionName="Tags">
  <AttributeList>
    <Name>MyIntegratedHmiTag_1</Name>
  </AttributeList>
  <LinkList>
    <AcquisitionCycle TargetID="@OpenLink">
      <Name>1 s</Name>
    </AcquisitionCycle>
    <Connection TargetID="@OpenLink">
      <Name>HMI_Connection_MP277_300400</Name>      <- Must exist in the project
    </Connection>
    <ControllerTag TargetID="@OpenLink">
      <Name>Datablock_1.DBElement1</Name>      <- Must exist in the project
    </ControllerTag>
  </LinkList>
</Hmi.Tag.Tag>
```

Hinweis

Die Zuweisung von Strukturvariablen in den Eigenschaften Obere und Untere Grenze der 'Elemente von Strukturvariablen' wird bei Comfort-/Advanced-Geräten nicht unterstützt. Diese Strukturvariablen können den Grenzeigenschaften von einfachen Variablen zugewiesen werden, werden aber nicht für die Grenzeigenschaften von Strukturvariablen-Elementen unterstützt.

Besondere Überlegungen für den Export/Import einer HMI-Variable des Datentyps "UDT"

Die Verknüpfung wird zum Datentyp exportiert, wenn eine HMI-Variable des Datentyps "UDT" exportiert wird. Nur versionierte Datentypen werden für den Import unterstützt.

Die Datentypen müssen in der Projektbibliothek gespeichert werden. Datentypen in der globalen Bibliothek werden nicht unterstützt.

Die folgenden Regeln gelten beim Import:

- Der referenzierte Datentyp muss in der Projektbibliothek enthalten sein.
Der Import wird beendet, wenn der Datentyp nicht in der Projektbibliothek enthalten ist.
 - Der referenzierte Datentyp muss versioniert sein. Versionierung wird ab TIA Portal V13 SP1 unterstützt.
Wenn der Datentyp nicht versioniert ist, wird eine Ausnahme ausgelöst.
-

Hinweis

Der erste gefundene Datentyp wird beim Import zum Auflösen der Referenz verwendet.

Hierbei gilt Folgendes: Zuerst wird das Wurzelverzeichnis der Projektbibliothek durchsucht, danach die Unterordner.

Verwenden Sie die folgende XML-Struktur, um eine HMI-Variable des Datentyps "UDT" zu importieren:

```
<Hmi.Tag.Tag ID="1" CompositionName="Tags">
  ...
  <LinkList>
    <DataType TargetID="@OpenLink">
      <Name>HmiUdt_1 V 1.0.0</Name>      <- Must exist in the project library
    </DataType>
    ...
  </LinkList>
  ...
</Hmi.Tag.Tag>
```

6.3.3 VB-Skripte

6.3.3.1 VB-Skripte exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Alle untergeordneten benutzerdefinierten Ordner werden beim Export berücksichtigt. Für jedes exportierte VB-Skript wird eine eigene XML-Datei erstellt.

Programmcode: VB-Skript exportieren

Um ein ausgewähltes VB-Skript eines HMI-Geräts in eine XML-Datei zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports a single vbscript of an HMI device
private static void ExportSingleVBScriptOfHMITarget(HmiTarget hmitarget)
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptComposition vbScripts = vbScriptFolder.VBScripts;
    VBScript vbScript = vbScripts.Find("MyVBScript");

    FileInfo info = new
    FileInfo(string.Format(@"C:\OpennessSamples\Export\Scripts\{0}.xml", vbScript.Name));
    vbScript.Export(info, ExportOptions.None);
}
```

6.3.3.2 VB-Skripte aus einem Ordner exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Für jedes exportierte VB-Skript wird eine eigene XML-Datei erstellt.

Programmcode: VB-Skript aus einem benutzerdefinierten Ordner exportieren

Um ein VB-Skript aus einem benutzerdefinierten Ordner in eine XML-Datei zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports vbscripts of a selected vbscript system folder
private static void ExportVBScriptOfSelectedVBScriptSystemFolder(HmiTarget hmitarget)
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptUserFolderComposition vbUserFolders = vbScriptFolder.Folders;
    VBScriptUserFolder vbUserFolder = vbUserFolders.Find("MyVBUserFolder");
    VBScriptComposition vbScripts = vbUserFolder.VBScripts;

    foreach (VBScript script in vbScripts)
    {
        FileInfo info = new
        FileInfo(String.Format(@"C:\OpennessSamples\Export\Scripts\{0}.xml", script.Name));
        script.Export(info, ExportOptions.None);
    }
}
```

Programmcode: Alle VB-Skripte aus einem Systemordner exportieren

Um alle VB-Skripte aus dem Systemordner zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all vbscripts by using a foreach loop
private static void ExportAllVBScripts(HmiTarget hmitarget)
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptComposition vbScripts = vbScriptFolder.VBScripts;
    if (vbScripts == null) return;

    foreach (VBScript script in vbScripts)
    {
        FileInfo info = new
        FileInfo(string.Format(@"C:\OpennessSamples\Export\Scripts\{0}.xml", script.Name));
        script.Export(info, ExportOptions.None);
    }
}
```

6.3.3.3 VB-Skripte importieren**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Sammelimporte werden unterstützt. Als Alternative können Sie einen Programmcode mit einer Foreach-Schleife verwenden (VB-Skripte exportieren (Seite 1279)).

Programmcode

Um ein VB-Skript aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;

namespace ImportVBScripts
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            ...
        }
        private static void ImportSingleVBScriptToHMITarget(HmiTarget hmitarget)
        {
            VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
            VBScriptComposition vbScripts = vbScriptFolder.VBScripts;
            if (vbScripts == null) return;
            {
                FileInfo info = new FileInfo(@"D:\Samples\Import\VBScript.xml");
                vbScripts.Import(info, ImportOptions.None);
            }
        }
    }
}
```

6.3.4 Textlisten

6.3.4.1 Textlisten aus einem HMI-Gerät exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Der Export von Text- und Grafiklisten umfasst alle Einträge der Listen. Text- und Grafiklisten können getrennt exportiert werden.

Die Textlisten eines HMI-Geräts werden exportiert. Für jede exportierte Textliste wird eine eigene XML-Datei erstellt.

Programmcode

Ändern Sie den folgenden Programmcode, um Textlisten aus einem HMI-Gerät zu exportieren:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;

namespace ExportTextListsFromHMIDevice
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            ...
        }
        //Export TextLists
        private static void ExportTextLists(HmiTarget hmitarget)
        {
            TextListComposition text = hmitarget.TextLists;
            foreach (TextList textList in text)
            {
                FileInfo info = new FileInfo(string.Format(@"D:\Samples\Export\{0}.xml", textList.Name));
                textList.Export(info, ExportOptions.WithDefaults);
            }
        }
    }
}
```

6.3.4.2 Textliste in ein HMI-Gerät importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die API-Schnittstelle unterstützt den Import einer Textliste aus einer XML-Datei in ein HMI-Gerät.

Programmcode

Um eine Textliste aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports a single TextList
private static void ImportSingleTextList(HmiTarget hmitarget)
{
    TextListComposition textListComposition = hmitarget.TextLists;
    IList<TextList> importedTextLists = textListComposition.Import(new
    FileInfo(@"D:\SamplesImport\myTextList.xml"), ImportOptions.Override);
}
```

6.3.4.3 Erweiterte XML-Formate für Export/Import von Textlisten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Standardexport von Textlisten
Siehe Textlisten aus einem Bediengerät exportieren (Seite 1282)
- Standardimport von Textlisten
Siehe Textlisten in ein Bediengerät exportieren (Seite 1284)

Verwendung

Eine Textliste kann auch formatierte Texte enthalten. Das betrifft hauptsächlich die folgende Formatierung:

- Textformatierung
- Verweise auf andere Objekte im Text

Reine Textformatierung in einer zu exportierenden Textliste führt zu einem erweiterten XML-Exportformat. Objektreferenzen werden als Open Links charakterisiert. Gleiches gilt für zu importierende Textlisten mit formatierten Texten.

Erweiterte XML-Exportformate können auch erheblich komplexer werden. Beispielsweise kann mehr als nur der Objektname in der Textliste verknüpft sein, möglicherweise durch ein Open Link zu einer PLC-Variable eines anderen Geräts. In diesem Fall müssen alle Informationen in einem String codiert werden, um das Open Link zu entfernen.

```

<?xml version="1.0" encoding="utf-8"?>
<Document>
<!-- ... -->
  <MultilingualText ID="5" CompositionName="Text">
    <ObjectList>
      <MultilingualTextItem ID="6" CompositionName="Items">
        <AttributeList>
          <Culture>en-US</Culture>
          <Text>
            <body>
              <p>
                <field ref="0" />
              </p>
            </body>
            <fieldinfos>
              <fieldinfo name="0" domaintype="HMICommonTextList">
                <reference TargetID="@OpenLink">
                  <name>Siemens.Simatic.Hmi.Utah.TextAndGraphicLists.HmiTextList:Empty Text_list_
                </reference>
                <subreference TargetID="@OpenLink">
                  <name>Siemens.Simatic.Hmi.Utah.Tag.HmiTag:t1</name>
                </subreference>
                <domaingroup>
                  <format length="9" />
                </domaingroup>
              </fieldinfo>
            </fieldinfos>
          </Text>
        </AttributeList>
      </MultilingualTextItem>
      <MultilingualTextItem ID="7" CompositionName="Items">
        <AttributeList>
          <Culture>de-CH</Culture>
          <Text>
            <body>
              <p>
                <field ref="0" />
              </p>
            </body>
            <fieldinfos>
              <fieldinfo name="0" domaintype="HMICommonTextList">
                <reference TargetID="@OpenLink">
                  <name>Siemens.Simatic.Hmi.Utah.TextAndGraphicLists.HmiTextList:Empty Text_list_
                </reference>
                <subreference TargetID="@OpenLink">
                  <name>Siemens.Simatic.Hmi.Utah.Tag.HmiTag:t1</name>
                </subreference>
                <domaingroup>
                  <format length="9" />
                </domaingroup>
              </fieldinfo>
            </fieldinfos>
          </Text>
        </AttributeList>
      </MultilingualTextItem>
    </ObjectList>
  </MultilingualText>

```

6.3.5 Grafiklisten

6.3.5.1 Grafiklisten exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Der Export von Text- und Grafiklisten umfasst alle Einträge der Listen. Text- und Grafiklisten können getrennt exportiert werden.

Pro Grafikliste wird eine XML-Datei erstellt. In den Grafiklisten enthaltene globale Grafikobjekte werden als Open Links exportiert.

Programmcode

Um Grafiklisten aus einem Bediengerät zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports GraphicLists
private static void ExportGraphicLists(HmiTarget hmitarget)
{
    GraphicListComposition graphic = hmitarget.GraphicLists;
    foreach (GraphicList graphicList in graphic)
    {
        FileInfo info = new FileInfo(string.Format(@"D:\Samples\Export\{0}.xml",
graphicList.Name));
        graphicList.Export(info, ExportOptions.WithDefaults);
    }
}
```

6.3.5.2 Grafikliste importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die API-Schnittstelle unterstützt den Import einer Grafikliste aus einer XML-Datei in ein HMI-Gerät.

Alle referenzierten Grafikobjekte der Grafikliste werden in den Import einbezogen. Referenzen auf globale Grafiken werden nicht einbezogen. Wenn die referenzierten globalen Grafiken im Zielprojekt vorhanden sind, werden die Referenzen auf die globalen Grafiken während des Imports wiederhergestellt.

Programmcode

Um eine Grafikliste aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
//Imports a single GraphicList
private static void ImportSingleGraphicList(HmiTarget hmitarget)
{
    GraphicListComposition graphicListComposition = hmitarget.GraphicLists;
    IList<GraphicList> importedGraphicLists = graphicListComposition.Import(new
    FileInfo(@"D:\Samples\Import\myGraphicList.xml"), ImportOptions.Override);
}
```

6.3.6 Verbindungen

6.3.6.1 Verbindungen exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Die API-Schnittstelle unterstützt den Export aller Verbindungen eines HMI-Geräts in eine XML-Datei.

Hinweis

Integrierte Verbindungen exportieren

Der Export integrierter Verbindungen wird nicht unterstützt.

Für jede exportierte Verbindung wird eine eigene XML-Datei erstellt.

Programmcode

Um alle Verbindungen eines HMI-Geräts in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports communication connections from an HMI device
private static void ExportConnectionsFromHMITarget(HmiTarget hmitarget)
{
    ConnectionComposition connections = hmitarget.Connections;
    foreach(Connection connection in connections)
    {
        FileInfo info = new FileInfo(string.Format(@"D:\Samples\Export\{0}.xml",
connection.Name));
        connexion.Export(info, ExportOptions.WithDefaults);
    }
}
```

6.3.6.2 Verbindungen importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Die API-Schnittstelle unterstützt den Import aller Verbindungen eines HMI-Geräts aus einer XML-Datei in ein HMI-Gerät. Wenn Sie mehrere Kommunikationsverbindungen importieren möchten, importieren Sie die jeweilige XML-Datei für die entsprechende Verbindung.

Hinweis

Wenn Sie eine Verbindung in ein Projekt importieren, in dem bereits eine integrierte Verbindung konfiguriert ist, wird diese Verbindung nicht überschrieben. Der Import wird abgebrochen und eine Exception wird ausgelöst.

Programmcode

Um eine einzelne Verbindung eines HMI-Geräts aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
//Imports Communication connections to an HMI device
private static void ImportConnectionsToHMITarget(HmiTarget hmitarget)
{
    ConnectionComposition connections = hmitarget.Connections;
    IList<Connection> importedConnectionLists = connections.Import(new
FileInfo(@"D:\Samples\Import\myConnectionImport.xml"), ImportOptions.Override);
}
```

6.3.7 Bilder

6.3.7.1 Übersicht der exportierbaren Bild-Objekte

Verwendung

Sie können die nachstehenden Bilder mit TIA Portal Openness APIs exportieren und importieren:

Tabelle 6-4 Unterstützte Bilder

Objekt	Export/Import möglich
Bild	Ja
Globales Bild	Ja
Bildvorlage	Ja
Permanentfenster	Ja
Pop-up-Bild	Ja
Slide-in-Bild	Ja

Sie können die nachstehenden Bildobjekte mit TIA Portal Openness APIs exportieren und importieren:

Tabelle 6-5 Unterstützte Bildobjekte

Bereich	Objektyp	Export/Import möglich
Grundlegende Objekte	Linie	Ja
	Polygonzug	Ja
	Polygon	Ja
	Ellipse	Ja
	Ellipsensegment	–
	Kreissegment	–
	Ellipsenbogen	–
	Kreisbogen	–
	Kreis	Ja
	Rechteck	Ja
	Verbinder	–
	Textfeld	Ja
	Grafikanzeige	Ja
	Rohr	–
	Doppel-T-Stück	–
	T-Stück	–
	Rohrkrümmer	–

Bereich	Objekttyp	Export/Import möglich
Elemente	EA-Feld	Ja
	Grafisches EA-Feld	Ja
	Editierbares Textfeld	–
	Listenfeld	–
	Kombinationsfeld	–
	Schaltfläche	Ja
	Rundbutton	–
	Leuchtdrucktaster	Ja
	Schalter	Ja
	Symbolisches EA-Feld	Ja
	Datum/Uhrzeit-Feld	Ja
	Balken	Ja
	Symbolbibliothek	Ja
	Schieberegler	Ja
	Bildlaufleiste	–
	Kontrollkästchen	–
	Optionsschaltfläche	–
	Zeigerinstrument	Ja
	Uhr	Ja
	Speicherplatzanzeige	–
	Funktionstasten (Softkeys)	Ja
	Gruppen	Ja
	Bildbausteininstanzen	Ja

Bereich	Objekttyp	Export/Import möglich
Bedienelemente	Bildfenster	–
	Benutzeranzeige	Ja
	Druckauftrag/Skriptdiagnose	–
	Kamera-Anzeige	–
	PDF-Anzeige	–
	Rezepturanzeige	–
	Meldeanzeige	–
	Meldeindikator	–
	Meldefenster	–
	f(x)-Kurvenanzeige	–
	f(t)-Kurvenanzeige	–
	Tabellenanzeige	–
	Wertetabelle	–
	HTML-Browser	–
	Media Player	–
	Kanaldiagnose	–
	WLAN-Empfang	–
	Zonen-Name	–
	Zonen-Signal	–
	Wirkbereichs-Name	–
	Wirkbereichs-Name (RFID)	–
	Wirkbereichs-Signal	–
	Ladezustand	–
	Handrad	–
	Hilfe-Indikator	–
	Sm@rtClient-Anzeige	–
	Status/Steuern	–
	Speicherplatzanzeige	–
	Anzeige von NC-Unterprogrammen	–
	Systemdiagnoseanzeige	–
	System-Diagnosefenster	–

Siehe auch

Grundlagen zum Import/Export (Seite 1249)

6.3.7.2 Alle Bilder eines HMI-Geräts exportieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Um alle aggregierten Bilder aller benutzerdefinierten Ordner eines HMI-Geräts zu exportieren, ist ein anderer Programmcode erforderlich.

Programmcode: Alle Bilder eines Geräts exportieren

Um die Bilder eines benutzerdefinierten Bilderordners eines HMI-Geräts und den Bildersystemordner zu exportieren, ändern Sie folgenden Programmcode:

6.3 Import/Export von Daten eines HMI-Geräts

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;

using Siemens.Engineering.SW.Blocks;

using Siemens.Engineering.SW.ExternalSources;

using Siemens.Engineering.SW.Tags;

using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;

namespace ExportAllScreensOfHMIDevice
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            ...
        }
        private static void ExportAllScreensOfDevice(string rootPath, HmiTarget hmiTarget)
        {
            DirectoryInfo info = new DirectoryInfo(rootPath);
            info.Create();
            // Export the ScreenFolder recursive
            string screenPath = Path.Combine(rootPath, "Screens");
            info = new DirectoryInfo(screenPath);
            info.Create();
            ExportScreens(screenPath, hmiTarget);
        }
        // Export the screens of a user-defined screen folder of an HMI device and the screen system
        // folder
        private static void ExportScreensOfDevice(HmiTarget hmiTarget)
        {
            ScreenUserFolder folder = hmiTarget.ScreenFolder.Folders.Find("MyScreenFolder");
            //or ScreenSystemFolder folder = hmiTarget.ScreenFolder;
            ScreenComposition screens = folder.Screens;
            foreach(Screen screen in screens)
            {

```

```
FileInfo info = new FileInfo(string.Format(@"D:\Samples\Screens\{0}\{1}.xml", folder.Name,
screen.Name));
screen.Export(info, ExportOptions.WithDefaults);
}
}
// Exporting all screens of a device independent of the user
public static void ExportScreens(string screenPath, HmiTarget target)
{
    foreach(Screen screen in target.ScreenFolder.Screens)
    {
        screen.Export(new FileInfo(Path.Combine(screenPath, screen.Name + ".xml")),
        ExportOptions.WithDefaults);
    }
    foreach(ScreenUserFolder subfolder in target.ScreenFolder.Folders)
    {
        ExportScreenUserFolder(Path.Combine(screenPath, folder.Name), subfolder.Name);
    }
}
private static void ExportScreenUserFolder(string screenPath,ScreenUserFolder folder )
{
    foreach(Screen screen in folder.Screens)
    {
        screen.Export(new FileInfo(Path.Combine(screenPath, screen.Name + ".xml")),
        ExportOptions.WithDefaults);
    }
    foreach(ScreenUserFolder subfolder in folder.Folders)
    {
        ExportScreenUserFolder(Path.Combine(screenPath, subfolder.Name), subfolder);
    }
}
}
```

6.3.7.3 Bild aus einem Bildordner exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten eines Bilds werden exportiert:

Bild	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name, Number, HelpText
Links öffnen	Template
Zusammensetzungen	- Layers - Animations Alle auf Runtime Advanced basierenden konfigurierten Animationen werden exportiert. - Events Alle auf Runtime Advanced basierenden konfigurierten Ereignisse werden exportiert. - Softkeys Alle konfigurierten Softkeys werden exportiert.

Für jede Schicht werden die folgenden Daten exportiert:

Hinweis

Standardmäßig besteht der Schichtname im TIA Portal aus leerem Text.

Wenn Sie den Schichtnamen im TIA Portal nicht ändern, ist der Name der exportierten Schicht leer. In diesem Fall ist der angezeigte Schichtname im TIA Portal von der Sprache der Benutzeroberfläche abhängig.

Wenn Sie den Schichtnamen im TIA Portal ändern, wird der geänderte Schichtname in allen entsprechenden Sprachen angezeigt.

Schicht	Daten
Attribute	Name, Index, VisibleES
Zusammensetzungen	ScreenItems (mit Bildelementen)

Nicht einbezogen werden in den Export:

- SCADA-spezifische Attribute
- Schichten, die keine Bildelemente enthalten und deren Attribute sich nicht von den Standardwerten unterscheiden.

Programmcode

Um ein einzelnes Bild aus dem Benutzerordner oder aus dem Systemordner eines HMI-Geräts zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a single screen from a screen folder
private static void ExportSingleScreenFromScreenFolder(HmiTarget hmitarget)
{
    ScreenUserFolder folder = hmitarget.ScreenFolder.Folders.Find("MyScreenFolder");
    //or ScreenSystemFolder folder = hmitarget.ScreenFolder;
    ScreenComposition screens = folder.Screens;
    Screen screen = screens.Find("Screen_1.xml");
    if (screen == null) return;
    {
        FileInfo info = new FileInfo(string.Format(@"D:\Samples\Screens\{0}\{1}.xml",
folder.Name, screen.Name));
        screen.Export(info, ExportOptions.WithDefaults);
    }
}
```

6.3.7.4 Bilder in ein HMI-Gerät importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die Bilder können nur in eine bestimmte Art von HMI-Gerät importiert werden. Das HMI-Gerät und das Gerät, aus dem die Bilder exportiert wurden, müssen den gleichen Gerätetyp besitzen.

Die folgenden Daten eines Bilds werden exportiert:

Bild	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name, Number, HelpText
Links öffnen	Templates
Zusammensetzungen	• Layers • Animations Alle für Bilder konfigurierbaren Animationen werden importiert. • Events Alle für Ereignisse konfigurierbaren Animationen werden importiert. • Softkeys Alle für Softkeys konfigurierbaren Animationen werden importiert.

Für jede Schicht werden die folgenden Daten importiert:

Hinweis

Wenn Sie vor dem Import für den Schichtnamen leeren Text angegeben haben, ist der angezeigte Schichtname im TIA Portal nach dem Import von der Sprache der Benutzeroberfläche abhängig.

Wenn Sie einen Schichtnamen zugewiesen haben, wird der angegebene Schichtname nach dem Import in allen entsprechenden Sprachen angezeigt.

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems

Einschränkungen

- Wenn die Breite und Höhe eines Bilds nicht den Abmessungen des Geräts entsprechen, wird der Importvorgang abgebrochen und eine Exception ausgelöst. Die Anpassung der enthaltenen Bildelemente wird nicht unterstützt. Aus diesem Grund können sich bestimmte Bildelemente außerhalb der Bildgrenzen befinden. In diesem Fall wird eine Compilerwarnung ausgegeben.
- Die Bildnummer muss für alle Bilder des Geräts jeweils eindeutig sein. Ein Bildimport wird abgebrochen, wenn ein Bild mit einer Bildnummer gefunden wird, die bereits im Gerät erstellt wurde. Wenn Sie noch keine Bildnummer zugewiesen haben, wird dem Bild während des Importvorgangs eine eindeutige Bildnummer zugeordnet.
- Die Anordnung der Bildelemente innerhalb der Z-Reihenfolge muss für jede Schicht im Bild eindeutig und lückenlos sein. Aus diesem Grund wird nach dem Import des Bilds eine Konsistenzprüfung durchgeführt, bei der die Anordnung gegebenenfalls repariert wird. Dieser Vorgang kann bei bestimmten Bildelementen zu geänderten „Tabindizes“ führen. Sie können die Z-Reihenfolge der Bildelemente in der XML-Datei manuell ändern. Das Bildelement an erster Stelle befindet sich ganz am Ende der Z-Reihenfolge.

Hinweis

Sie können die Werte für Breite und Höhe eines Bildelements in der XML-Datei ändern, wenn für das Bildelement das Attribut „Größe an Inhalt anpassen“ aktiviert ist.

Hinweis**Import von Bildtypen aus der Bibliothek wird nicht unterstützt**

Ab WinCC V12 SP1 können Sie in der Bibliothek ein Bild als Typ erstellen. Instanzen des im Projekt verwendeten Bildtyps können wie andere Bilder mit der TIA Portal Openness-Anwendung bearbeitet werden. Wenn Sie Bilder exportieren, werden die Instanzen von Bildtypen ohne die Typinformationen exportiert.

Wenn Sie diese Bilder wieder in das Projekt importieren, werden die Instanzen des Bildtyps überschrieben und die Instanz wird vom Bildtyp abgelöst.

Programmcode: Bilder in ein HMI-Gerät importieren

Um mit der For each-Schleife Bilder in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
//Imports all screens to an HMI device
private static void ImportScreensToHMITarget(HmiTarget hmitarget)
{
    FileInfo[] exportedScreens = new FileInfo[] {new
    FileInfo(@"D:\Samples\Import\Screen_1.xml"), new
    FileInfo(@"D:\Samples\Import\Screen_2.xml")};
    ScreenUserFolder folder = hmitarget.ScreenFolder.Folders.Find("MyScreenFolder");
    foreach (FileInfo screenFileInfo in exportedScreens)
    {
        folder.Screens.Import(screenFileInfo, ImportOptions.Override);
    }
}
```

Programmcode: In einen neu erstellten Benutzerordner importieren

Um ein Bild in einen neu erstellten Benutzerordner eines HMI-Geräts zu importieren, ändern Sie folgenden Programmcode:

```
//Imports a single screen to a new created user folder of an HMI device
private static void ImportSingleScreenToNewFolderOfHMITarget(HmiTarget hmitarget)
{
    ScreenUserFolder folder = hmitarget.ScreenFolder.Folders.Create("MyFolder");
    folder.Screens.Import(new FileInfo(@"D:\Samples\Import\myScreens.xml"),
    ImportOptions.Override);
}
```

6.3.7.5 Permanentfenster exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten des Permanentfensters werden exportiert:

Permanentfenster	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name
Zusammensetzungen	Layers

Für jede Schicht werden die folgenden Daten exportiert:

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems (mit Bildelementen)

Programmcode

Um ein Permanentfenster eines HMI-Geräts in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a permanent area
private static void ExportScreenoverview(HmiTarget hmitarget)
{
    ScreenOverview overview = hmitarget.ScreenOverview;
    if (overview == null) return;

    FileInfo info = new FileInfo(@"D:\Samples\Screens\ExportedOverview.xml");
    overview.Export(info, ExportOptions.WithDefaults);
}
```

6.3.7.6 Permanentfenster importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten des Permanentfensters werden importiert:

Permanentfenster	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name, Visible, Number
Zusammensetzungen	Layers

Für jede Schicht werden die folgenden Daten importiert:

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems (mit Bildelementen)

Wenn die Breite und Höhe eines Bilds nicht den Abmessungen des Geräts entsprechen, wird der Importvorgang abgebrochen und eine Exception ausgelöst. Die Anpassung der enthaltenen Geräteelemente (Bildelemente) wird nicht unterstützt. Aus diesem Grund können sich einige Geräteelemente außerhalb der Bildgrenzen befinden. In diesem Fall wird eine Compilerwarnung ausgegeben.

Die Anordnung der Geräteelemente im Permanentfenster muss eindeutig und lückenlos sein. Aus diesem Grund wird nach dem Import des Permanentfensters eine Konsistenzprüfung durchgeführt, bei der die Anordnung gegebenenfalls repariert wird. Dieser Vorgang kann bei bestimmten Geräteelementen zu geänderten „Tabindizes“ führen.

Programmcode

Um ein Permanentfenster aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
//Imports a permanent area
private static void ImportScreenOverview(HmiTarget hmiTarget)
{
    FileInfo info = new FileInfo(@"D:\Samples\Screens\ExportedOverview.xml");
    hmiTarget.ImportScreenOverview(info, ImportOptions.Override);
}
```

6.3.7.7 Alle Bildvorlagen eines HMI-Geräts exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Pro Bildvorlage wird eine XML-Datei erstellt.

6.3 Import/Export von Daten eines HMI-Geräts

Da Sammelexporte nicht unterstützt werden, müssen Sie alle Bildvorlagen enumerieren und separat exportieren. Achten Sie dabei darauf, dass die verwendeten Bildvorlagennamen den Dateibenennungskonventionen Ihres Dateisystems entsprechen.

Programmcode: Alle Bildvorlagen eines Geräts exportieren

Um alle Bildvorlagen aus einem bestimmten Ordner zu exportieren, ändern Sie folgenden Programmcode:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;

namespace ExportAllScreenTemplatesOfHMIDevice
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            ...
        }
        public static void ExportScreenTemplatesOfDevice(string rootPath, ScreenTemplateUserFolder
            folder)
        {
            string screenPath = Path.Combine(rootPath, "Screens");
            DirectoryInfo info = new DirectoryInfo(screenPath);
            info.Create();
            //Export the ScreenTemplateFolder recursive
            ExportScreenTemplates (screenPath, hmiTarget);
        }
        //Exports all screen templates of a selected folder
        private static void ExportScreenTemplates(string templatePath, HmiTarget hmiTarget)
        {
            foreach (ScreenTemplate screen in hmiTarget.ScreenTemplateFolder.ScreenTemplates)
            {
                screen.Export(new FileInfo(Path.Combine(templatePath, screen.Name + ".xml")),
                    ExportOptions.WithDefaults);
            }
            foreach (ScreenTemplateUserFolder folder in hmiTarget.ScreenTemplateFolder.Folders)
            {
                ExportScreenTemplates(Path.Combine(templatePath, folder.Name), hmiTarget);
            }
        }
    }
}
```

}

6.3.7.8 Bildvorlagen aus einem Ordner exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten der Bildvorlage werden exportiert:

Bildvorlagen	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name
Zusammensetzungen	• Layers • Animations Alle konfigurierten Animationen werden exportiert. SCADA-Animationen werden nicht exportiert. • Softkeys Alle konfigurierten Softkeys werden exportiert.

Für jede Schicht werden die folgenden Daten exportiert:

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems (mit Bildelementen)

Programmcode: Bildvorlage eines benutzerdefinierten Ordners exportieren

Um eine einzelne Bildvorlage aus dem Systemordner oder einem benutzerdefinierten Ordner zu exportieren, ändern Sie folgenden Programmcode:


```

using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;

namespace ExportScreenTemplatesOfFolder
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            ...
        }
        private static void ExportSingleScreenTemplate(string templatePath, HmiTarget hmiTarget)
        {
            ScreenTemplateUserFolder folder =
            hmiTarget.ScreenTemplateFolder.Folders.Find("MyTemplateFolder");
            //or ScreenTemplateSystemFolder folder = hmiTarget.ScreenTemplateFolder;
            ScreenTemplateComposition templates = folder.ScreenTemplates;
            ScreenTemplate template = templates.Find("templateName");
            if(template == null) return;
            FileInfo info = new FileInfo(string.Format(@"D:\Samples\Templates\{0}\{1}.xml",
            folder.Name, template.Name));
            template.Export(info, ExportOptions.WithDefaults);
        }
        // Exporting all screen templates of a user-defined folder
        public static void ExportScreenTemplateUserFolder(string rootPath,
        ScreenTemplateUserFolder folder)
        {
            DirectoryInfo info = new DirectoryInfo(rootPath);
            info.Create();
            foreach (ScreenTemplate screen in folder.ScreenTemplates)
            {
                screen.Export(new FileInfo(Path.Combine(info.FullName, screen.Name + ".xml")),
                ExportOptions.WithDefaults);
            }
        }
    }
}

```

```

foreach (ScreenTemplateUserFolder subfolder in folder.Folders)
{
ExportScreenTemplateUserFolder(Path.Combine(info.FullName, subfolder.Name), subfolder);
}
}
// Exports all screen templates of a selected folder
private static void ExportScreenTemplates(string templatePath, ScreenTemplateUserFolder
folder)
{
foreach (ScreenTemplate screen in folder.ScreenTemplates)
{
screen.Export(new FileInfo(Path.Combine(templatePath, screen.Name + ".xml")),
ExportOptions.WithDefaults);
}
foreach (ScreenTemplateUserFolder subfolder in folder.Folders)
{
ExportScreenTemplates(Path.Combine(templatePath, subfolder.Name), subfolder);
}
}
}
}

```

6.3.7.9 Bildvorlagen importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Die folgenden Daten einer Bildvorlage werden importiert:

Bildvorlage	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name, SetTabOrderInFront
Zusammensetzungen	• Layers • Animations Alle für Bilder konfigurierbaren Animationen werden importiert. • Softkeys Alle für Softkeys konfigurierbaren Animationen werden importiert.

Für jede Schicht werden die folgenden Daten importiert:

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems (mit Bildelementen)

Wenn die Breite und Höhe einer Bildvorlage nicht den Abmessungen des Geräts entsprechen, wird der Importvorgang abgebrochen und eine Exception ausgelöst. Die Anpassung der enthaltenen Bildelemente wird nicht unterstützt. Aus diesem Grund können sich bestimmte Bildelemente außerhalb der Bildgrenzen befinden. In diesem Fall wird eine Compilerwarnung ausgegeben.

Die Anordnung der Geräteelemente in der Bildvorlage muss eindeutig und lückenlos sein. Aus diesem Grund wird nach dem Import der Bildvorlage eine Konsistenzprüfung durchgeführt, bei der die Anordnung gegebenenfalls repariert wird. Dieser Vorgang kann bei bestimmten Bildelementen zu geänderten „Tabindizes“ führen.

Programmcode: Allgemeiner Import

Um mit der For each-Schleife alle Bildvorlagen in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```

using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;

namespace ImportingScreenTemplates
{
    internal class Program
    {
        //Imports screen templates to an HMI device
        private static void ImportScreenTemplatesToHMITarget(HmiTarget hmiTarget)
        {
            ScreenTemplateUserFolder folder =
            hmiTarget.ScreenTemplateFolder.Folders.Find("MyTemplateFolder");
            // or ScreenTemplateSystemFolder folder = hmiTarget.ScreenTemplateFolder;
            FileInfo[] exportedTemplates = new FileInfo[] {new
            FileInfo(@"D:\Samples\Import\Template_1.xml"), new
            FileInfo(@"D:\Samples\Import\Template_n.xml")};
            foreach (FileInfo templateFileName in exportedTemplates)
            {
                folder.ScreenTemplates.Import(templateFileName, ImportOptions.Override);
            }
        }
        //Imports screen templates to a user folder of an HMI device
        private static void ImportScreenTemplatesToFolderOfHMITarget(HmiTarget hmiTarget)
        {
            ScreenTemplateUserFolder screenTemplateFolder =
            hmiTarget.ScreenTemplateFolder.Folders.Find("MyTemplateFolder");
            ScreenTemplateUserFolder folder = screenTemplateFolder.Folders.Create("MyNewFolder");
            folder.ScreenTemplates.Import(new FileInfo(@"D:\Samples\Import\ScreenTemplate.xml"),
            ImportOptions.Override);
        }
    }
}

```

6.3.7.10 Exportieren eines Pop-up-Bilds

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Die folgenden Daten des Pop-Up-Bilds werden exportiert:

Bildvorlagen	Daten
Attribute	ActiveLayer, BackColor, GridColor, Height, Name, ScrollbarBackgroundColor, ScrollbarForegroundColor, Width
Zusammensetzungen	• Layers • Events Alle konfigurierten Ereignisse werden exportiert.

Für jede Schicht werden die folgenden Daten exportiert:

Schicht	Daten
Attribute	Name, Index, VisibleES
Zusammensetzungen	ScreenItems Alle exportierbaren Bildobjekte werden exportiert.

Programmcode: Exportieren eines Pop-up-Bilds aus einem Ordner

Um ein einzelnes Pop-up-Bild aus dem Systemordner oder einem benutzerdefinierten Ordner zu exportieren, ändern Sie folgenden Programmcode:

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;

namespace ExportingAPopupImage
{
    internal class Program
    {
        //Exports a single pop-up screen
        private static void ExportSinglePopUpScreen(HmiTarget hmitarget)
        {
            ScreenPopupUserFolder folder = hmitarget.ScreenPopupFolder.Folders.Find("MyPopupFolder");
            //or ScreenPopupSystemFolder folder = hmitarget.ScreenPopupFolder;
            ScreenPopupComposition popups = folder.ScreenPopups;
            ScreenPopup popup = popups.Find("popupName");
            if(popup == null) return;
            FileInfo info = new FileInfo(string.Format(@"D:\Samples\Screens\{0}\{1}.xml", folder.Name,
            popup.Name));
            popup.Export(info, ExportOptions.WithDefaults);
        }
    }
}
```

6.3.7.11 Importieren eines Pop-up-Bildes

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten eines Pop-up-Bilds werden importiert:

Bildvorlagen	Daten
Attribute	ActiveLayer, BackColor, GridColor, Height, Name, ScrollbarBackgroundColor, ScrollbarForegroundColor, Width
Zusammensetzungen	• Layers • Events Alle konfigurierten Ereignisse werden exportiert.

Die Existenz folgender Attribute ist für das Importieren verpflichtend:

- Name
- Height
- Width

Für jede Schicht werden die folgenden Daten importiert:

Schicht	Daten
Attribute	Name, Index, VisibleES
Zusammensetzungen	ScreenItems Alle importierbaren Bildobjekte werden importiert.

Einschränkungen

Wenn das Gerät keine Pop-up-Bilder unterstützt, wird der Kopiervorgang abgebrochen und eine Ausnahme ausgelöst.

Wenn die Breite und Höhe eines Pop-up-Bilds nicht den Abmessungen des Geräts entsprechen, wird der Importvorgang abgebrochen und eine Exception ausgelöst.

- Minimale Höhe = 1 Pixel
- Minimale Breite = 1 Pixel
- Max. Höhe = sechsfache Höhe des Gerätebildschirms
- Max. Breite = zweifache Breite des Gerätebildschirms
- Bei Geräten mit Runtime-Version V13 SP1 entspricht die maximale Höhe und die maximale Breite der Höhe und Breite des Gerätebildschirms.

Programmcode: Importieren eines Pop-up-Bilds in einen Ordner

Um ein Pop-up-Bild in einen Pop-up-Bild-Systemordner oder in einen benutzerdefinierten Ordner zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports a pop-up screen to an HMI device
private static void ImportPopupScreenToHMITarget(HmiTarget hmitarget)
{
    FileInfo info = new FileInfo(string.Format(@"D:\Samples\Screens\PopupScreen.xml"));
    hmitarget.ScreenPopupFolder.ScreenPopups.Import(info, ImportOptions.None);
}
```

6.3.7.12 Exportieren eines Slide-in-Bilds

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten und Werte des Slide-in-Bilds werden exportiert:

Bildvorlagen	Daten	
Attribute	Activate	false
	ActiveLayer	0
	BackColor	(182; 182; 182)
	GridColor	(0; 0; 0)
	Dimension	427 Das Attribut "Dimension" gibt entweder die Breite oder die Höhe des Slide-in-Bilds an, abhängig von der Art des Slide-in-Bilds.
	LineColor1	(223; 223; 223)
	LineColor2	(32; 32; 32)
	OperatableAreaColor	(128; 128; 128)
	SlideinType	Oben, Unten, Links, Rechts Slide-in-Bilder haben keinen Namen, sondern einen Slidein-Type.
	Visibility	FadeOut
Zusammensetzungen	Layers	

Hinweis

Slide-in-Bilder haben keinen Namen, sondern einen SlideinType.

Für jede Schicht werden die folgenden Daten exportiert:

Schicht	Daten	
Attribute	Name,	
	Index	
	VisibleES	
Zusammensetzungen	ScreenItems	Alle exportierbaren Bildobjekte werden exportiert.

Programmcode: Exportieren eines Slide-in-Bilds

Um ein einzelnes Slide-in-Bild aus dem Systemordner zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a single slide-in screen
private static void ExportSingleSlideinScreen(HmiTarget hmitarget)
{
    ScreenSlideinSystemFolder systemFolder = hmitarget.ScreenSlideinFolder;
    var screens = systemFolder.ScreenSlideins;
    ScreenSlidein slidein = screens.Find(SlideinType.Bottom);
    if (slidein == null) return;

    FileInfo info = new FileInfo(string.Format(@"D:\Samples\Screens\{0}\{1}.xml"));
    slidein.Export(info, ExportOptions.WithDefaults);
}
```

6.3.7.13 Importieren eines Slide-in-Bilds**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten und Werte eines Slide-in-Bilds werden importiert:

Bildvorlagen	Daten
Attribute	Activate = false ActiveLayer = 0 Authorization BackColor = (182; 182; 182) Dimension = 427 Das Attribut „Dimension“ gibt entweder die Breite oder die Höhe des Slide-in-Bilds an, abhängig davon, welches der zwei Attribute für die Art des Slide-in-Bilds geändert werden kann. GridColor = (0; 0; 0) LineColor1 = (223; 223; 223) LineColor2 = (32; 32; 32) OperateableAreaColor = (128; 128; 128) SlideinType = Top, Bottom, Left, Right Visibility = FadeOut
Zusammensetzungen	Layers

Die Existenz des folgenden Attributs ist für das Importieren verpflichtend:

- SlideinType

Für jede Schicht werden die folgenden Daten importiert:

Schicht	Daten
Attribute	Name, Index, VisibleES
Zusammensetzungen	ScreenItems Alle importierbaren Bildobjekte werden importiert.

Einschränkungen

- Wenn das Gerät keine Slide-in-Bilder unterstützt, wird der Import abgebrochen und eine Ausnahme ausgelöst.
- Wird ein Slide-in-Bild von einem anderen Element referenziert, ist das Slide-in-Bild über openlink zu referenzieren und nicht über SlideinType, z. B. in der Systemfunktion „ShowSlideinScreen“).

Die folgende Tabelle zeigt die Abbildung des Attributs "SlideinType" mit dem entsprechenden openlink:

SlideinType	Openlink-Name
Top	GraphX_Slidein_Top
Right	GraphX_Slidein_Right
Bottom	GraphX_Slidein_Bottom
Left	GraphX_Slidein_Left

Programmcode: Importieren eines Slide-in-Bilds in einen Ordner

Um ein Slide-in-Bild in einen Slide-in-Bild-Systemordner zu importieren, ändern Sie folgenden Programmcode:

```
//Imports a slide-in screen to an HMI device
private static void ImportSlideinScreenToHMITarget(HmiTarget hmitarget)
{
    FileInfo info = new FileInfo(@"D:\Samples\Screens\SlideInScreen.xml");
    hmitarget.ScreenSlideinFolder.ScreenSlideins.Import(info, ImportOptions.None);
}
```

6.3.7.14 Bild mit Eingabemaske exportieren**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten einer Bildbausteininstanz in einem Bild werden exportiert:

Bild	Daten
Attribute	Left, Top, Width, Height, ObjectName, Resizing, TabIndex, FaceplateTypeName
Schnittstellenattribute	Alle konfigurierten Schnittstellenattribute einer Bildbausteininstanz werden als exportierbare Bildelemente exportiert.
Zusammensetzungen	• Animationen Alle bewegten Animationen werden exportiert. Variablenanimationen basieren auf den Attributen der Schnittstelle. • Ereignisse Alle konfigurierten Ereignisse werden exportiert.

Beachten Sie die folgenden Vorgaben für exportierte Attribute von Bildbausteininstanzen:

- **Resizing**
Das Attribut „Resizing“ wird in jedem Fall exportiert, unabhängig von den Exportoptionen.
- **FaceplateTypeName**
Das Attribut "FaceplateTypeName" identifiziert den entsprechenden Typ und die Version des Bildbausteins, z. B. „Faceplate_1 V 0.0.2“.

Hinweis

Bildbausteintyp in einem Bibliotheksordner

Befindet sich ein Bildbausteintyp in einem Bibliotheksordner, benötigen Sie den gesamten Pfad und Namen zur Identifizierung des Bildbausteintyps. Das Schlüsselwort „@\$@“ wird verwendet, um Ordner und/oder Namen von Bildbausteintypen zu trennen, z. B. „Folder_1@\$@SubFolder_1@\$@Faceplate_1 V 0.0.2“.

Die folgenden Daten von Bildelementen innerhalb einer Bildbausteininstanz sind vom Export ausgeschlossen:

Bildelement	Attribut
E/A-Feld	Flashing on limit violation
Grafisches E/A-Feld	Fit embedded graphic object to screen size

Programmcode

Um ein einzelnes Bild mit einer Bildbausteininstanz zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports a single screen including a faceplate instance
private static void ExportSingleScreenWithFaceplateInstance(HmiTarget hmitarget)
{
    ScreenFolder folder = hmitarget.ScreenFolder.Folders.Find("MyScreenFolder");
    ScreenComposition screens = folder.Screens;
    Screen screen = screens.Find("ScreenWithFaceplateName");
    if (screen == null) return;
    {
        FileInfo info = new FileInfo(string.Format(@"D:\Samples\Faceplates\{0}\{1}.xml",
        folder.Name, screen.Name));
        screen.Export(info, ExportOptions.WithDefaults);
    }
}
```

6.3.7.15 Bild mit Eingabemaske importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Die folgenden Daten einer Bildbausteininstanz in einem Bild werden importiert:

Bild	Daten
Attribute	Left, Top, Width, Height, ObjectName, Resizing, TabIndex, FaceplateTypeName
Schnittstellenattribute	Alle konfigurierten Schnittstellenattribute einer Bildbausteininstanz werden als importierbare Bildelemente importiert.
Zusammensetzungen	• Animationen Alle bewegten Animationen werden importiert. Variablenanimationen basieren auf den Attributen der Schnittstelle. • Ereignisse Alle konfigurierten Ereignisse werden importiert.

Die Existenz folgender Attribute ist für das Importieren verpflichtend:

- ObjectName
- FaceplateTypeName

Die folgenden Daten von Bildelementen innerhalb einer Bildbausteininstanz sind von Export und Import ausgeschlossen:

Bildelement	Attribut
E/A-Feld	Flashing on limit violation
Grafisches E/A-Feld	Fit embedded graphic object to screen size

Einschränkungen

- Bildbaustein, Ereignis oder Schnittstellenattribut unbekannt
Wenn ein faceplate type name, ein event name oder ein interface attribute name in der Importdatei angegeben ist, der nicht im Projekt existiert, wird der Import abgebrochen und eine Ausnahme ausgegeben.
- Skalierungsverhalten einer Bildbausteininstanz
Das Attribut „Resizing“ wird in jedem Fall importiert, unabhängig von den Exportoptionen.
Beispiele:
Ist "Resizing" auf "KeepRatio" gesetzt, dann wird das Attribut "Height" verwendet, um den Attributwert "Width" zu berechnen.
 - Die Größe eines Bildbausteintyps beträgt 100 x 100 Pixel. Wenn eine Bildbausteininstanz mit einer Größe von 300 x 100 Pixel importiert wird und der Wert "FixedSize" auf das Attribut "Resizing" eingestellt ist, ist der Import erfolgreich und die Bildbausteingröße wird auf 100 x 100 Pixel gesetzt.
 - Die Größe eines Bildbausteintyps beträgt 100 x 50 Pixel. Wenn eine Bildbausteininstanz mit einer Größe von 100 x 100 Pixel importiert wird und der Wert "KeepRatio" auf das Attribut "Resizing" eingestellt ist, ist der Import erfolgreich und die Bildbausteingröße wird auf 200 x 100 Pixel gesetzt.

Hinweis

Skalierungsverhalten von importierten Bildbausteininstanzen

Die Werte von „Resizing“ und die Werte der Schnittstellenattribute können die Größe der importierten Bildbausteininstanz und sogar die Größe der darin enthaltenen Bildelemente beeinflussen.

Um ungewünschte Änderungen am Aussehen einer Bildbausteininstanz zu vermeiden, importieren Sie einen Bildbaustein in der ursprünglichen Größe oder ohne die Attributwerte "Width" und "Height".

- Abweichende Schnittstellenattributwerte
 - Wenn Sie die Attribute für den Import ändern, wird der letzte verwendete Schnittstellenattributwert importiert.
 - Wenn die Attribute voneinander abhängig sind, können andere Attribute während des Imports geändert werden.
Beispiel: Ein Bildbaustein enthält ein E/A-Feld. Das Attribut „Betriebsart“ wird mit einem Schnittstellenattribut verbunden. Wenn Sie erst die Betriebsart auf „Ausgang“ und dann das Attribut „Versteckter Eingang“ auf wahr setzen, wird der Wert von „Versteckter Eingang“ nach dem Import nicht verwendet. Das Attribut „Versteckter Eingang“ wurde bei der ersten Änderung auf schreibgeschützt gesetzt und der Wert kann nicht verwendet werden.
 - Wenn ein Attributwert die Einschränkungen von WinCC nicht erfüllt, wird der Wert des Bildbausteintyps angezeigt.
Beispiel: Der Anzeigebereich eines Zeigerinstrument wird auf 10 - 80 eingestellt. Die Attribute "Maximalwert" und "Minimalwert" werden als Schnittstellenattribute konfiguriert. Wenn Sie einen Minimalwert einstellen, der den Maximalwert übersteigt, z. B. 100, dann wird der Wert des Bildbausteintyps für „Minimalwert“ nach dem Import angezeigt.

- Wenn ein Schnittstellenattribut mit mehreren Attributen von Bildelementen innerhalb des Bildbausteintyps verbunden ist, dann zeigt das Schnittstellenattribut an der Bildbausteininstanz den verwendeten Attributwert des ersten verbundenen Bildelements an.
Beispiel: Ein Bildbaustein enthält zwei Zeigerinstrumente mit abweichenden Maximalwerten. Die Minimalwerte beider Zeigerinstrumente werden mit einem einzelnen Schnittstellenattribut verbunden.
Wenn Sie erst einen Minimalwert einstellen, der für beide Zeigerinstrumente anwendbar ist, werden beide Werte eingestellt.
Wenn Sie dann einen Wert einstellen, der nur für das zweite Zeigerinstrument anwendbar ist, wird der Wert nur für das zweite Zeigerinstrument eingestellt, der Wert des ersten Zeigerinstruments wird hingegen als Schnittstellenattribut angezeigt.

Programmcode: Bilder mit einer Bildbausteininstanz importieren

Um ein Bild mit einer Bildbausteininstanz zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports single screen including a faceplate instance
private static void ImportSingleScreenWithFaceplateInstance(HmiTarget hmitarget)
{
    FileInfo info = new FileInfo(@"D:\Samples\Screens\ScreenFaceplate.xml");
    hmitarget.ScreenFolder.Screens.Import(info, ImportOptions.None);
}
```

6.4 Daten eines PLC-Geräts importieren/exportieren

6.4.1 CFC-Pläne (Export/Import)

6.4.1.1 Export/Import von CFC-Plänen

Die TIA Portal Openness API unterstützt den Export und Import von mit STEP 7 CFC ("Continuous Function Chart") erstellten Plänen.

Sie können alle oder nur ausgewählte Pläne als XML-Dateien exportieren und importieren.

Sie rufen diese Funktionen für definierte Aufgaben außerhalb des TIA Portal über die Public API auf.

Hinweis

Empfohlene PC-Hardware

Wenn Sie mit großen Projekten arbeiten, prüfen Sie, ob der Computer die Hardwareanforderungen des TIA Portals erfüllt:

- RAM:
32 GB bei großen Projekten
-

Funktionen

Folgende Funktionen stehen für CFC-Pläne zur Verfügung:

- Export von CFC-Plänen (Seite 1328)
- Export nur von ausgewählten CFC-Plänen (Seite 1329)
- Import von CFC-Plänen (Seite 1331)
- Passwörter für Pläne konfigurieren:
 - Passwort für CFC-Pläne festlegen (Seite 1332)
 - Passwort aus CFC-Plänen lesen (Seite 1333)
 - Passwort für CFC-Pläne ändern (Seite 1335)
 - Passwort aus CFC-Plänen löschen (Seite 1336)

Die Funktionsbeschreibungen umfassen Codebeispiele, die Sie für Ihr Openness-Programm anpassen können.

Sicherheitsmaßnahmen für TIA Portal Openness-Anwendungen

Es ist empfehlenswert

- eine TIA Portal Openness-Anwendung mit Administratorrechten im Ordner "Programme" zu installieren.
- das dynamische Laden von Programmbestandteilen wie Assemblies oder DLLs aus dem Benutzerbereich zu vermeiden.
- die TIA Portal Openness-Anwendung mit Benutzerrechten auszuführen.

Hinweis

Siemens ist nicht verantwortlich für die Kompatibilität von Daten und Informationen, die über diese Schnittstellen mit Fremdsoftware übertragen werden, und übernimmt dafür keine Gewähr.

Wir weisen ausdrücklich darauf hin, dass eine unsachgemäße Nutzung der Schnittstellen zu Datenverlust und zum Produktionsstillstand führen kann.

Hinweis

Im Zusammenhang mit dieser Beschreibung gibt es keinerlei Verpflichtungen oder Garantien, die Quelldatei manuell zu ändern und auszuwerten.

Siemens übernimmt deshalb keine Haftung für den Gebrauch dieser Beschreibung als Ganzes oder in Teilen.

Wenn Sie extern erzeugte Konfigurationsdaten importieren, die Codefehler, eine falsche Struktur oder unerwünschte Manipulationen enthalten, kann dies unerwartete Fehler und Sicherheitsrisiken verursachen.

**WARNUNG**

Der API-Benutzer ist verantwortlich für die Sicherheitsmaßnahmen zur Handhabung von Passwörtern durch Code.

Weitere Informationen finden Sie der Dokumentation für TIA Portal Openness:

- "Liesmich TIA Portal Openness"
- "Grundlagen > Aufgaben von Openness > Einführung (Seite 50)"
- "TIA Portal Openness API > Allgemeine Funktionen > TIA Portal Openness Firewall (Seite 90)"
- "TIA Portal Openness API > Funktionen für den Zugriff auf die Daten eines PLC-Geräts > Funktionen zum Laden von Daten ins PLC-Gerät > Herunterladen zu den PLCs (Seite 400)"
- "Export/import > Überblick > Grundlagen zum Export/Import (Seite 1249)"

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

TIA Portal-Projektansicht: CFC-Pläne exportieren und importieren (Seite 1326)

6.4.1.2 TIA Portal-Projektansicht: CFC-Pläne exportieren und importieren

Im TIA Portal-Projekt können Sie CFC-Pläne über den Dialog "Export / Import CFC" exportieren bzw. importieren.

Der Dialog verwendet die Openness-Funktionen zum Export bzw. Import von CFC-Plänen. Die gleichen Randbedingungen gelten wie beim direkten Einsatz der Openness-Funktionen.

Befolgen Sie die Hinweise unter "Export/Import von CFC-Plänen (Seite 1324)" im Abschnitt "Sicherheitsmaßnahmen für TIA Portal Openness-Anwendungen".

Datentransfer-Dialog

Verwenden Sie für den Import den Dialog "Datentransfer - Generieren/Importieren"

Weitere Informationen zu diesem Dialog:

- Industry Online Support: "SIMATIC Prozessleitsystem PCS 7 Hilfe zum Datentransfer-Dialog" (<https://support.industry.siemens.com/cs/ww/de/view/109812471>)

Voraussetzungen

- "TIA Openness" ist installiert.
- Die PLC ist nicht online.

- Die zu exportierenden CFC-Pläne sind nicht passwortgeschützt.
CFC-Pläne mit konfigurierten Passwörtern werden beim Export ignoriert.
- Import: Die importierten Bausteintypen sind unter der PLC im TIA Portal angelegt und übersetzt.
Wenn die importierte XML-Datei Bausteine enthält, die noch nicht angelegt sind, wird der Import abgebrochen.

Vorgehen

1. Wählen Sie in der Projektnavigation den Eintrag "Pläne" oder einen CFC-Plan.
2. Wählen Sie in der Menüleiste den Eintrag "Werkzeuge > Import / Export CFC".
Der Dialog "Export / Import CFC" öffnet sich.
3. Wählen Sie die gewünschte Aktion:
 - Alle CFC-Pläne der PLC exportieren
Mit den CFC-Plänen werden auch die Bausteintypen sowie die Einstellungen für Tasks und Ablaufreihenfolge exportiert.
 - Einzelne CFC-Pläne exportieren
Die CFC-Pläne werden ohne Bausteintypen, Task-Einstellungen und Ablaufreihenfolge exportiert.
Klicken Sie auf "Next" und wählen Sie die gewünschten Pläne.
 - CFC-Pläne importieren
Den Umfang des Imports wählen Sie im folgenden Dialog "Datentransfer - Generieren/Importieren".
4. Klicken Sie auf "Next".
5. Wählen Sie den Ablagepfad der XML-Datei und klicken Sie auf "Finish".
Der Export bzw. Import wird gestartet.
6. Wenn Sie CFC-Pläne exportieren, bestätigen Sie die Meldung, dass die XML-Datei geschrieben wurde.
Die XML-Datei mit den exportierten Daten liegt im gewählten Ablagepfad.
7. Wenn Sie CFC-Pläne importieren, öffnet sich der Dialog "Datentransfer - Generieren/Importieren".
Wählen Sie die zu importierenden Objekte und klicken Sie im Bereich "Generieren/Importieren" auf die Schaltfläche "Objekte importieren von B nach A": 
Die gewünschten Daten werden unter der PLC ins TIA Portal-Projekt importiert.
Das Inspektorfenster enthält gegebenenfalls Hinweise zum Import.

Siehe auch

Export von CFC-Plänen (Seite 1328)

Export nur von ausgewählten CFC-Plänen (Seite 1329)

Import von CFC-Plänen (Seite 1331)

6.4.1.3 Export von CFC-Plänen

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe "Verbindung zum TIA Portal herstellen (Seite 84)"
- Ein Projekt ist geöffnet.
Siehe "Öffnen eines Projekts (Seite 131)"
- Die PLC ist nicht online.
- Die zu exportierenden CFC-Pläne sind nicht passwortgeschützt.
CFC-Pläne mit konfiguriertem Passwort werden beim Export ignoriert.

Verwendung

Die TIA Portal Openness API unterstützt den Export von CFC-Plänen in eine XML-Datei mit der Funktion "CompleteExport".

Die Funktion schreibt die CFC-Projektdateien aus dem Planordner in eine XML-Datei:

- Alle CFC-Pläne, die unter der ausgewählten PLC erstellt wurden
- In den exportierten Plänen verwendete Bausteintypen
- Aufgabenzuweisungen für die exportierten Pläne
- Ablaufreihenfolge jedes exportierten Plans

Weitere Informationen zu den unterstützten Objekten finden Sie in der CFC-Dokumentation: "Anweisungen und Bausteine".

Wenn Sie einen selektiven XML-Export von CFC-Plänen starten wollen, müssen Sie die Funktion "SelectiveExport (Seite 1329)" verwenden.

Parameter

Parameter	Datentyp	Beschreibung
xmlFilePath	String	Ordnerpfad und Name der Importdatei
modelVersion	String	Zu verwendende Version des S7TIA-Austauschmodells
filter	Int64	Filteroptionen für die Automatisierungsschnittstelle
unattended	Boolean	Trigger für Hintergrundmodus

Programmcode

Ändern Sie den folgenden Programmcode, um alle CFC-Pläne einer PLC mit ihren Objekten in eine XML-Datei zu exportieren.

```
plcSoftware = (PlcSoftware) swContainer.Software;

chartProvider = plcSoftware.GetService<ChartProviderS7>();
if (chartProvider == null)    // in case that CFC is not installed
    return;

// Export CFC charts:
// XML file information for export
chartProvider.CompleteExport(@"D:\Users\username1\Documents\Automation\OpennessExample.xml", "V2.0", 0, true);
```

Weitere Informationen finden Sie der Dokumentation für TIA Portal Openness:

- "Export/Import > Überblick > Konfigurationsdaten exportieren (Seite 1253)"

Siehe auch

Export/Import von CFC-Plänen (Seite 1324)

TIA Portal-Projektansicht: CFC-Pläne exportieren und importieren (Seite 1326)

6.4.1.4 Export nur von ausgewählten CFC-Plänen

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe "Verbindung zum TIA Portal herstellen (Seite 84)"
- Ein Projekt ist geöffnet.
Siehe "Öffnen eines Projekts (Seite 131)"
- Die PLC ist nicht online.
- Die zu exportierenden CFC-Pläne sind nicht passwortgeschützt.
CFC-Pläne mit konfiguriertem Passwort werden beim Export ignoriert.

Verwendung

Die TIA Portal Openness API unterstützt den Export von spezifischen CFC-Plänen in eine XML-Datei mit der Funktion "SelectiveExport".

Die Funktion schreibt die Projektdaten nur für die ausgewählten Pläne in eine XML-Datei:

- Spezifische CFC-Pläne aus der ausgewählten PLC
- In den exportierten Plänen verwendete Bausteintypen
- Aufgabenzuweisungen für die exportierten Pläne
- Ablaufreihenfolge jedes exportierten Plans

Weitere Informationen zu den unterstützten Objekten finden Sie in der CFC-Dokumentation: "Anweisungen und Bausteine".

Wenn Sie einen vollständigen XML-Export von allen CFC-Plänen starten wollen, müssen Sie die Funktion "CompleteExport (Seite 1328)" verwenden.

Parameter

Parameter	Datentyp	Beschreibung
xmlFilePath	String	Ordnerpfad und Name der Importdatei
selectedObjects	String[]	Namen der Pläne, die Sie exportieren wollen
modelVersion	String	Zu verwendende Version des S7TIA-Austauschmodells
filter	Int64	Filteroptionen für die Automatisierungsschnittstelle
unattended	Boolean	Trigger für Hintergrundmodus

Programmcode

Ändern Sie den folgenden Programmcode, um nur ausgewählte CFC-Pläne mit ihren Objekten in eine XML-Datei zu exportieren.

```
plcSoftware = (PlcSoftware) swContainer.Software;

chartProvider = plcSoftware.GetService<ChartProviderS7>();
if (chartProvider == null)    // in case that CFC is not installed
    return;

// Export selected CFC charts
// XML file information for export of CFC charts "CFC_1" and "CFC_3"
chartProvider.SelectiveExport(@"D:\Users\username1\Documents\Automation\OpennessExample.xml", new string[] { "CFC_1", "CFC_3" }, "V2.0", 0, true);
```

Weitere Informationen finden Sie der Dokumentation für TIA Portal Openness:

- "Export/Import > Überblick > Konfigurationsdaten exportieren (Seite 1253)"

Siehe auch

Export/Import von CFC-Plänen (Seite 1324)

TIA Portal-Projektansicht: CFC-Pläne exportieren und importieren (Seite 1326)

6.4.1.5 Import von CFC-Plänen

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe "Verbindung zum TIA Portal herstellen (Seite 84)"
- Ein Projekt ist geöffnet.
Siehe "Öffnen eines Projekts (Seite 131)"
- Die PLC ist nicht online.

Verwendung

Die TIA Portal Openness API unterstützt den Import von CFC-Plänen aus einer XML-Datei mit der Funktion "Import".

Die XML-Datei wird durch einen vollständigen XML-Export oder einen selektiven XML-Export nur bestimmter CFC-Pläne erstellt.

Parameter

Parameter	Datentyp	Beschreibung
xmlFilePath	String	Ordnerpfad und Name der Importdatei
modelVersion	String	Zu verwendende Version des S7TIA-Austauschmodells
filter	Int64	Filteroptionen für die Automatisierungsschnittstelle Der Parameter wird in der aktuellen CFC-Version beim Export bzw. Import nicht ausgewertet und hat keine Funktion.
unattended	Boolean	Trigger für Hintergrundmodus
deleteAtTarget	Boolean	Trigger zum Löschen von Objekten im TIA-Projekt, die in der ursprünglichen Exportdatei nicht enthalten waren

Programmcode

Um CFC-Pläne aus einer XML-Datei zu importieren, ändern Sie folgenden Programmcode:

```
plcSoftware = (PlcSoftware) swContainer.Software;

chartProvider = plcSoftware.GetService<ChartProviderS7>();
if (chartProvider == null)    // in case that CFC is not installed
    return;

// Import CFC charts
// XML file information for import
chartProvider.Import(@"D:\Users\username1\Documents\Automation\OpennessExample.xml",
"V2.0", 0, true, false);
```

Weitere Informationen finden Sie der Dokumentation für TIA Portal Openness:

- "Export/Import > Überblick > Konfigurationsdaten importieren (Seite 1255)"

Siehe auch

Export/Import von CFC-Plänen (Seite 1324)

TIA Portal-Projektansicht: CFC-Pläne exportieren und importieren (Seite 1326)

6.4.1.6 Passwort für CFC-Pläne festlegen**Voraussetzungen**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe "Verbindung zum TIA Portal herstellen (Seite 84)"
- Ein Projekt ist geöffnet.
Siehe "Öffnen eines Projekts (Seite 131)"
- Die PLC ist nicht online.

Verwendung

Um einen CFC-Plan oder einen hierarchischen CFC-Plan vor unbeabsichtigter Bearbeitung zu schützen, können Sie für den Plan ein Passwort konfigurieren.

- Verwenden Sie zum Konfigurieren des Passworts die Funktion "AddChartProtection".
- Verwenden Sie zum Ändern eines bestehenden Passworts die Funktion "ChangeChartProtection (Seite 1335)".
- Verwenden Sie zum Lesen des Passworts als Hash-Wert aus einem Plan die Funktion "GetChartProtection (Seite 1333)".
- Verwenden Sie zum Löschen des konfigurierten Passworts die Funktion "RemoveChartProtection (Seite 1336)".

ACHTUNG**Passwort: Keine Autorisierung, kein Know-how-Schutz der Pläne**

Das Passwort dient nur dem Schutz vor unbeabsichtigtem Bearbeiten eines CFC-Plans.
Diese Form des Zugriffsschutzes dient nicht dazu, die Zugriffssicherheit zu erhöhen.

Was das Passwort nicht bietet:

- Schutz vor unbefugtem Zugriff auf Know-how in CFC-Plänen
- Security-relevante Autorisierung für den Zugriff auf CFC-Pläne

Parameter

Parameter	Datentyp	Beschreibung
chartName	System.String	Name des mit einem Passwort geschützten Plans
newHashedPassword	System.String	Passwort Das Passwort wird als Hash-Wert angezeigt.

Rückgabewert

Die Funktion "AddChartProtection" gibt einen System.Boolean zurück:

TRUE	Das Passwort wurde erfolgreich festgelegt.
FALSE	Das Passwort konnte nicht festgelegt werden.

Programmcode

Ändern Sie folgenden Programmcode, um ein Passwort für einen CFC-Plan festzulegen.

In diesem Beispiel ist der Hash-Wert des neuen Passworts als abgekürzter Beispielswert enthalten.

```
plcSoftware = (PlcSoftware) swContainer.Software;  
  
chartProvider = plcSoftware.GetService<ChartProviderS7>();  
if (chartProvider == null)    // in case that CFC is not installed  
    return;  
  
// Configure CFC chart password  
bool added = chartProvider.AddChartProtection("CFC_1", "AgGUWq...92M=");
```

Siehe auch

Export/Import von CFC-Plänen (Seite 1324)

6.4.1.7 Passwort aus CFC-Plänen lesen**Voraussetzungen**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe "Verbindung zum TIA Portal herstellen (Seite 84)"
- Ein Projekt ist geöffnet.
Siehe "Öffnen eines Projekts (Seite 131)"
- Die PLC ist nicht online.

Verwendung

Verwenden Sie zum Lesen des Passworts aus einem CFC-Plan die Funktion "GetChartProtection".

Sie können das Passwort mit den Funktionen "ChangeChartProtection (Seite 1335)" und "RemoveChartProtection (Seite 1336)" ändern oder löschen.

ACHTUNG**Passwort: Keine Autorisierung, kein Know-how-Schutz der Pläne**

Das Passwort dient nur dem Schutz vor unbeabsichtigtem Bearbeiten eines CFC-Plans.

Diese Form des Zugriffsschutzes dient nicht dazu, die Zugriffssicherheit zu erhöhen.

Was das Passwort nicht bietet:

- Schutz vor unbefugtem Zugriff auf Know-how in CFC-Plänen
- Security-relevante Autorisierung für den Zugriff auf CFC-Pläne

Parameter

Parameter	Datentyp	Beschreibung
chartName	System.String	Name des mit einem Passwort geschützten Plans

Rückgabewert

Die Funktion "GetChartProtection" gibt einen System.Boolean zurück:

TRUE	Das Passwort wurde erfolgreich gelesen.
FALSE	Das Passwort konnte nicht gelesen werden.

Programmcode

Ändern Sie folgenden Programmcode, um ein Passwort aus einem CFC-Plan auszulesen.

```
plcSoftware = (PlcSoftware) swContainer.Software;

chartProvider = plcSoftware.GetService<ChartProviderS7>();
if (chartProvider == null)    // in case that CFC is not installed
    return;

// Read CFC chart password
string passwordHash = chartProvider.GetChartProtection("CFC_1");
```

Siehe auch

Passwort für CFC-Pläne festlegen (Seite 1332)

Export/Import von CFC-Plänen (Seite 1324)

6.4.1.8 Passwort für CFC-Pläne ändern

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe "Verbindung zum TIA Portal herstellen (Seite 84)"
- Ein Projekt ist geöffnet.
Siehe "Öffnen eines Projekts (Seite 131)"
- Die PLC ist nicht online.

Verwendung

Um einen CFC-Plan oder einen hierarchischen CFC-Plan durch Ändern des Passworts vor unbeabsichtigter Bearbeitung zu schützen, verwenden Sie die Funktion "ChangeChartProtection".

Sie können ein Passwort mit den Funktionen "GetChartProtection (Seite 1333)" und "RemoveChartProtection (Seite 1336)" lesen oder löschen.

ACHTUNG

Passwort: Keine Autorisierung, kein Know-how-Schutz der Pläne

Das Passwort dient nur dem Schutz vor unbeabsichtigtem Bearbeiten eines CFC-Plans. Diese Form des Zugriffsschutzes dient nicht dazu, die Zugriffssicherheit zu erhöhen.

Was das Passwort nicht bietet:

- Schutz vor unbefugtem Zugriff auf Know-how in CFC-Plänen
- Security-relevante Autorisierung für den Zugriff auf CFC-Pläne

Parameter

Parameter	Datentyp	Beschreibung
chartName	System.String	Name des mit einem Passwort geschützten Plans
currentPassword	System.Security.SecureString	Aktuell für den CFC-Plan verwendetes Passwort
newHashedPassword	System.String	Neues Passwort Das Passwort wird als Hash-Wert angezeigt.

Rückgabewert

Die Funktion "ChangeChartProtection" gibt einen System.Boolean zurück:

TRUE	Das Passwort wurde erfolgreich geändert.
FALSE	Das Passwort konnte nicht geändert werden.

Programmcode

Ändern Sie folgenden Programmcode, um ein Passwort für einen CFC-Plan zu ändern.

6.4 Daten eines PLC-Geräts importieren/exportieren

In diesem Beispiel wird das Passwort "test" in ein neues Passwort geändert. Der Hash-Wert des neuen Passworts ist als abgekürzter Beispielwert enthalten.

```
plcSoftware = (PlcSoftware) swContainer.Software;

chartProvider = plcSoftware.GetService<ChartProviderS7>();
if (chartProvider == null)    // in case that CFC is not installed
    return;

// Change CFC chart password
char[] password1 = new char[] {'t', 'e', 's', 't'};

SecureString securePassword1 = new SecureString();
foreach (char ch in password1)
    securePassword1.AppendChar(ch);

bool changed = chartProvider.ChangeChartProtection("CFC_1", securePassword1,
"AgGUWq...92M=");
```

Siehe auch

- Passwort für CFC-Pläne festlegen (Seite 1332)
- Export/Import von CFC-Plänen (Seite 1324)

6.4.1.9 Passwort aus CFC-Plänen löschen

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe "Verbindung zum TIA Portal herstellen (Seite 84)"
- Ein Projekt ist geöffnet.
Siehe "Öffnen eines Projekts (Seite 131)"
- Die PLC ist nicht online.

Verwendung

Um das konfigurierte Passwort eines CFC-Plans zu löschen, verwenden Sie die Funktion "RemoveChartProtection".

Der CFC-Plan kann ohne Eingeben des Passworts wieder geöffnet und bearbeitet werden.

Verwenden Sie zum Festlegen eines neuen Passworts die Funktion "AddChartProtection (Seite 1332)".

ACHTUNG

Passwort: Keine Autorisierung, kein Know-how-Schutz der Pläne

Das Passwort dient nur dem Schutz vor unbeabsichtigtem Bearbeiten eines CFC-Plans.

Diese Form des Zugriffsschutzes dient nicht dazu, die Zugriffssicherheit zu erhöhen.

Was das Passwort nicht bietet:

- Schutz vor unbefugtem Zugriff auf Know-how in CFC-Plänen
- Security-relevante Autorisierung für den Zugriff auf CFC-Pläne

Parameter

Parameter	Datentyp	Beschreibung
chartName	System.String	Name des mit einem Passwort geschützten Plans
currentPassword	System.Security.SecureString	Aktuell für den CFC-Plan verwendetes Passwort

Rückgabewert

Die Funktion "RemoveChartProtection" gibt einen System.Boolean zurück:

TRUE	Das Passwort wurde erfolgreich gelöscht.
FALSE	Das Passwort konnte nicht gelöscht werden.

Programmcode

Ändern Sie folgenden Programmcode, um ein Passwort für einen CFC-Plan zu löschen.

In diesem Beispiel wurde für den Plan "CFC_1" das Passwort "test" konfiguriert.

```
plcSoftware = (PlcSoftware) swContainer.Software;

chartProvider = plcSoftware.GetService<ChartProviders7>();
if (chartProvider == null)    // in case that CFC is not installed
    return;

// Remove CFC chart password
char[] password1 = new char[] {'t', 'e', 's', 't'};

SecureString securePassword1 = new SecureString();
foreach (char ch in password1)
    securePassword1.AppendChar(ch);

bool removed = chartProvider.RemoveChartProtection("CFC_1", securePassword1);
```

Siehe auch

Passwort für CFC-Pläne ändern (Seite 1335)

Passwort aus CFC-Plänen lesen (Seite 1333)

Export/Import von CFC-Plänen (Seite 1324)

6.4.2 Bausteine

6.4.2.1 XML-Struktur des Abschnitts Bausteinschnittstelle

Grundprinzip

Die Daten in der Exportdatei vom Import/Export sind mit Verweis auf eine Grundstruktur gegliedert. Jede Importdatei muss die grundlegenden strukturellen Bedingungen erfüllen.

Die Exportdatei enthält alle bearbeiteten Variablen und Konstanten des Schnittstellenabschnitts eines exportierten Bausteins. Alle Attribute mit `"ReadOnly = \"TRUE\""` und `"Informative = \"TRUE\""` werden ausgeschlossen.

Wenn die Information redundant ist, muss sie in der Import-XML-Datei und der Projektdatei identisch sein. Andernfalls wird beim Import eine wiederherstellbare Ausnahme ausgelöst.

Die Projektdaten enthalten mehr Daten als die Import-XML-Datei, z. B. kann ein externer Typ zusätzliche Mitglieder haben.

Nur schreibbare Werte können über TIA Portal Openness XML importiert werden.

Abhängig von den Exporteinstellungen in TIA Portal Openness enthält die Exportdatei einen festgelegten Satz von Attributen und Elementen. Die aus höheren Versionen des Produkts exportierte XML ist beim Importvorgang in niedrigeren Versionen des TIA Portals nicht kompatibel.

Grundstruktur

Der Schnittstellenabschnitt eines exportierten Bausteins ist im Element `<Interface>` in der SimaticML eines Bausteins enthalten. Das Stammobjekt ist das Element `<Sections>`, das den Schnittstellenabschnitt eines exportierten Bausteins darstellt. Die Reihenfolge der folgenden Beschreibung von Elementen stellt die erforderliche Reihenfolge in der Eingabedatei dar.

```
<Interface>
  <Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v1">
    <Section Name="Input">
      <Member Name="input1" Datatype="Bool" Remanence="Volatile" Accessibility="Public">
        <AttributeList>
          ...
        </AttributeList>
      </Member>
      <Member Name="input2" Datatype="Bool" Remanence="Volatile" Accessibility="Public">
        <AttributeList>
          ...
        </AttributeList>
      </Member>
    </Section>
    <Section Name="Output">
      <Member Name="output1" Datatype="Bool" Remanence="Volatile" Accessibility="Public">
        <AttributeList>
          ...
        </AttributeList>
      </Member>
    </Section>
    <Section Name="InOut" />
    <Section Name="Static" />
    <Section Name="Temp" />
    <Section Name="Constant" />
  </Sections>
</Interface>
```

- Abschnitt
Abschnitt stellt einen einzelnen Parameter oder Lokaldaten eines Programmbausteins dar.
- Mitglied
Mitglied stellt die im Programmbaustein verwendeten Variablen oder Konstanten dar. Abhängig vom Datentyp einer Variablen können Mitglieder verschachtelt sein oder weitere strukturelle Unterelemente haben.
Beim Datentyp "ARRAY" stellt das strukturelle Element "Subelement Path" z. B. den Index von Komponenten eines Array-Elements dar.
Es werden nur die Mitglieder exportiert, die vom Anwender bearbeitet wurden.
- AttributeList
Die `<AttributeList>` umfasst alle definierten Attribute eines Mitglieds. Attribute, die vom System festgelegt oder von einem Standardwert zugewiesen sind, werden in der XML-Struktur nicht aufgeführt.
Die Mitgliedsattribute `<ReadOnly>` und `<Informative>` werden nur in die XML-Exportdatei geschrieben, wenn ihr Wert TRUE ist.

- StartValue
Das Element <StartValue> wird nur geschrieben, wenn der Standardwert der Variablen oder Konstanten vom Anwender festgelegt ist.

```
...
<Member Name="Static_1" Datatype="Int">
...
  <StartValue>1</StartValue>
</Member>
...
```

- Kommentar
Das Element <Comment> wird geschrieben, wenn es vom Anwender festgelegt ist.
Kommentare einer Variablen oder Konstanten werden als mehrsprachiger Text exportiert:

```
...
<Member Name="Static_3" Datatype="Struct">
  <AttributeList>
    <BooleanAttribute Name="ExternalAccessible" SystemDefined="true">false</BooleanAttribute>
    <BooleanAttribute Name="ExternalVisible" SystemDefined="true">false</BooleanAttribute>
    <BooleanAttribute Name="ExternalWritable" SystemDefined="true">false</BooleanAttribute>
  </AttributeList>
  <Comment>
    <MultiLanguageText Lang="de-DE">An individual comment</MultiLanguageText>
  </Comment>
</Member>
...
```


Attribute

- Hauptattribute
Die Hauptattribute werden in das Element <Member> in der XML-Struktur geschrieben.

```
...
<Member Name="Static_1" Datatype="User_data_type_1" Remanence="Retain">
...
</Member>
...
```

Die folgende Tabelle zeigt die Hauptattribute einer Variablen oder Konstanten am Abschnitt Bausteinschnittstelle.

Name	Datentyp	Standardeinstellung	Importbedingung	Kommentar
Name	STRING	-	Erforderlich	
Datentyp	ENUM	-	Erforderlich	
Version	STRING	-	Optional	
Remanenz	ENUM	NonRetain	-	Wird nur geschrieben, wenn es sich nicht um die Standardeinstellung handelt
Zugänglichkeit	ENUM	Öffentlich	-	Vom System vordefiniert Kann vom Anwender nicht geändert werden
Informativ	BOOL	FALSE	-	

Mitglieder mit dem Merker "Informative" werden beim Import ignoriert. Wenn das Attribut gelöscht oder auf FALSE gesetzt ist, wird eine Ausnahme ausgelöst.

Hinweis

Remanenzeinstellungen "Im IDB setzen"

Wenn der Remanenzwert einer Variablen oder Konstanten "Im IDB setzen" ist, muss die im IDB festgelegte Remanenz für alle anderen Variablen und Konstanten mit dem Remanenzwert "SetInIDB" die gleiche sein.

Das erste importierte Mitglied mit dem Attribut "Im IDB setzen" legt die erwartete Remanenz im IDB für die folgenden Variablen und Konstanten mit dem Remanenzwert "SetInIDB" fest.

- Systemdefinierte Mitgliedsattribute
Systemdefinierte Mitgliedsattribute werden im Element <AttributeList> aufgelistet. Systemdefinierte Mitgliedsattribute werden mit dem Merker <Informative> gekennzeichnet und beim Import ignoriert.

6.4 Daten eines PLC-Geräts importieren/exportieren

```

...
<Member Name="Static_3" Datatype="Struct">
  <AttributeList>
    <BooleanAttribute Name="ExternalAccessible" SystemDefined="true">false</BooleanAttribute>
    <BooleanAttribute Name="ExternalVisible" SystemDefined="true">false</BooleanAttribute>
    <BooleanAttribute Name="ExternalWritable" SystemDefined="true">false</BooleanAttribute>
  </AttributeList>
  <Comment>
    <MultiLanguageText Lang="de-DE">An individual comment</MultiLanguageText>
  </Comment>
</Member>
...

```

Name	Typ	Standardeinstellung	SimaticMLschreibgeschützt (informativ)	Kommentar
At	String	""	FALSE	Mitgliedsanteile sind mit einem anderen Mitglied in dieser Struktur versetzt
SetPoint	Bool	FALSE	FALSE	Mitglied kann mit Arbeitsspeicher synchronisiert werden
UserReadOnly	Bool	FALSE	TRUE	Anwender kann keine Mitgliedsattribute ändern (einschl. Name)
UserDeletable	Bool	TRUE	TRUE	Mitglied kann im Editor nicht gelöscht werden
HmiAccessible	Bool	TRUE	FALSE	Kein HMI-Zugriff, kein Strukturelement
HmiVisible	Bool	TRUE	FALSE	Filter zum Verringern der Anzahl Mitglieder, die zuerst angezeigt werden
Offset	Int	-	TRUE	DB, FB, FC (Temp). Bei klassischen PLCs und Plus PLCs, wo die klassische Remanenz eingestellt ist.
PaddedSize	Int	-	TRUE	DB, FB, FC (Temp). Bei klassischen PLCs und Plus PLCs, wo die klassische Remanenz eingestellt ist. Nur bei Arrays.

Name	Typ	Standardeinstellung	SimaticML schreibgeschützt (informativ)	Kommentar
HiddenAssignment	Bool	FALSE	FALSE	Zuweisung bei Aufruf ausblenden, wenn sie PredefinedAssignment entspricht.
PredefinedAssignment	String	""	FALSE	Eingabe für den verwendeten Parameter, wenn der Aufruf platziert wird.
ReadOnlyAssignment	Bool	FALSE	FALSE	Der Anwender kann die vordefinierte Zuweisung beim Aufruf nicht ändern.
UserVisible	Bool	TRUE	TRUE	Dieses Mitglied wird in der UI nicht angezeigt.
HmiReadOnly	Bool	TRUE	TRUE	Dieses Mitglied für die HMI schreibgeschützt.
CodeReadOnly	Bool	FALSE	TRUE	-

- Anwenderdefinierte Attribute**
 Anwenderdefinierte Attribute werden mit dem Merker <ReadOnly> gekennzeichnet. Mitglieder mit diesem Merker werden beim Import ignoriert. Wenn der Merker gelöscht oder auf FALSE gesetzt ist, wird eine Ausnahme ausgelöst.
 Nicht bearbeitete anwenderdefinierte Attribute werden vom Export ausgeschlossen.

Name	Typ	Standardeinstellung	SimaticML schreibgeschützt (informativ)	Kommentar
CFC	IBlockAttribute	---	FALSE	Dies ist eine Ladung

Datentyp "STRUCT"

Die Komponenten des Datentyps "STRUCT" werden in der XML-Struktur einer Import-/Exportdatei als verschachtelte Mitglieder dargestellt:

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype="Struct">
      <!-- Basic struct -->
      <Member Name="Static_1" Datatype="Int">
        <!-- First Member of struct -->
        <StartValue>1</StartValue>
      </Member>
      <Member Name="Static_2" Datatype="Int">
        <!-- Second Member of struct -->
      </Member>
      <Member Name="Static_3" Datatype="Struct">
        <!-- A subsequent struct -->
        <Member Name="Static_1" Datatype="Int">
          <!-- First Member of the subsequent struct -->
          <StartValue>3</StartValue>
        </Member>
        <Member Name="Static_2" Datatype="Int">
          <!-- Second Member of the subsequent struct -->
        </Member>
      </Member>
    </Member>
  </Section>
</Sections>
```

Datentyp "ARRAY" Basistyp

Die Komponenten des Basisdatentyps "ARRAY" werden in der XML-Struktur einer Import-/Exportdatei als Unterelemente mit dem Attribut "Path" dargestellt:

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype="Array[0..2] of Int" Remanence="Retain">
      <!-- Basic Array -->
      <Subelement Path="0">
        <!-- First Array Component-->
        <StartValue>1</StartValue>
      </Subelement>
      <Subelement Path="1">
        <!-- Second Array Component-->
        <StartValue>2</StartValue>
      </Subelement>
    </Member>
  </Section>
</Sections>
```

Datentyp "ARRAY" von UDT

Die Komponenten des Datentyps "ARRAY" eines UDT werden in der XML-Struktur einer Import-/Exportdatei als neues Element `<sections>` in einem Element `<member>` dargestellt. Die Mitglieder im neuen Abschnitt für UDT in einem ARRAY sind als Unterelemente mit dem Attribut "Path" zugewiesen:

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype=""User_data_type_1"" Retainence="Retain">
      <Sections> <!-- Sections including the UDT "User_data_type_1" -->
        <Section Name="None">
          <Member Name="Element_2" Datatype="Int">
            <StartValue>47</StartValue>
          </Member>
        </Section>
      </Sections>
    </Member>
    <Member Name="Static_2" Datatype="Array[0..1] of "User_data_type_1"">
      <Sections> <!-- Sections including the UDT "User_data_type_1" -->
        <Section Name="None">
          <Member Name="Element_2" Datatype="Int">
            <Subelement Path="0"> <!-- Component of the array -->
              <StartValue>123</StartValue>
            </Subelement>
          </Member>
        </Section>
      </Sections>
    </Member>
  </Section>
</Sections>
```

Datentyp "ARRAY" in "ARRAY"

Die Komponenten des Datentyps "ARRAY" in einem anderen ARRAY werden in der XML-Struktur einer Import-/Exportdatei als Unterelemente mit dem Attribut "Path" dargestellt.

Die Mitglieder in einem anderen ARRAY werden als Unterelemente mit dem Attribut "Path" zugewiesen, wenn die Komponente vom Anwender bearbeitet wird:

```

<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype="Array[0..2] of Struct">
      <Member Name="Static_1" Datatype="Int" />
      <Member Name="Static_2" Datatype="Array[0..1, 0..3, -9..-2] of Struct">
        <Member Name="Static_1" Datatype="Int">
          <Subelement Path="0,0,3,-5">
            <StartValue>1</StartValue>
          </Subelement>
        </Member>
        <Subelement Path="0,0,2,-6">
          <Comment>
            <MultiLanguageText Lang="de-DE">A individual comment</MultiLanguageText>
          </Comment>
        </Subelement>
      </Member>
    </Member>
  </Section>
</Sections>

```

PLC-Datentypen (UDT)

Die XML-Struktur eines PLC-Datentyps ist von den Exporteinstellungen in TIA Portal Openness abhängig.

- `ExportOptions.None`
Mitglieder des PLC-Datentyps werden nur geschrieben, wenn der Standardwert von mindestens einer der Komponenten vom Anwender festgelegt ist. Bei diesen Mitgliedern werden nur die zwei zusätzlichen Attribute "Name" und "Datatype" geschrieben, um das Mitglied zu identifizieren, zu dem der `<StartValue>` gehört. Andere Mitglieder und Attribute werden nicht geschrieben.
- `ExportOptions.WithDefaults`
Die folgenden Attribute werden immer geschrieben:
 - Name
 - Datatype
 - ExternalAccessible
 - ExternalVisible
 - ExternalWritable
 - SetPoint
 - StartValue
Sie werden nur in XML geschrieben, wenn der Standardwert in diesem Typ vom Anwender festgelegt ist. Wenn er nur im PLC-Datentyp festgelegt ist, wird er nicht geschrieben.
- `ExportOptions.ReadOnly`
Bei PLC-Datentypen führt diese Einstellung nicht zu einem aussagekräftigen Ergebnis. In Kombination mit anderen Einstellungen hat dies keine Auswirkung auf das Ergebnis.

Überlagerte Variablen

Wenn eine Variable mit einem neuen Datentyp überlagert wird, werden die Mitglieder in der XML-Struktur des neuen Datentyps dargestellt. Die folgende XML-Struktur zeigt einen Datentyp WORD, der mit einem ARRAY von BYTE überlagert ist.

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Input" />
  <Section Name="Output" />
  <Section Name="InOut" />
  <Section Name="Static">
    <Member Name="Static_1" Datatype="Word">
      <!-- Basic Member -->
      <StartValue>16#17</StartValue>
    </Member>
    <Member Name="Static_2" Datatype="Array[0..1] of Byte">
      <AttributeList>
        <StringAttribute Name="At" SystemDefined="true">Static_1</StringAttribute>
      </AttributeList>
      <!-- The AT member -->
      <Subelement Path="0">
        <!-- First overlay byte -->
      </Subelement>
      <Subelement Path="1">
        <!-- Second overlay byte -->
      </Subelement>
    </Member>
  </Section>
  <Section Name="Temp" />
  <Section Name="Constant" />
</Sections>
```

Bausteinschnittstelle

Alle Attribute mit `ReadOnly = "TRUE"` und `Informative = "FALSE"` werden ausgeschlossen. Die XML-Struktur einer Bausteinschnittstelle ist von den Exporteinstellungen in TIA Portal Openness abhängig.

- `ExportOptions.None`
Diese Einstellung exportiert nur die geänderten Daten oder die Daten, die von den Standardwerten abweichen.
Wenn die Attributdefinition keinen Standardwert angibt, wird das Attribut immer geschrieben.
Die Exportdatei enthält auch alle Werte, die für den anschließenden Datenimport obligatorisch sind.

- `ExportOptions.WithDefaults`
Die folgenden Attribute werden immer geschrieben:

- `Name`
- `Datatype`
- `HmiAccessible` exportiert als `ExternalAccessible`
- `HmiVisible` exportiert als `ExternalVisible`
- `ExternalWritable`
- `SetPoint` (sofern vorhanden)
- `Offset` (sofern vorhanden)
- `PaddedSize` (sofern vorhanden)

Alle anderen Attribute werden nur geschrieben, wenn sich ihre Werte vom Standardwert unterscheiden.

Das Element `<StartValue>` wird nur in XML geschrieben, wenn es explizit festgelegt wurde.

- `ExportOptions.ReadOnly`
Bei Bausteinschnittstellen führt diese Einstellung nicht zu einem aussagekräftigen Ergebnis. In Kombination mit anderen Einstellungen hat dies keine Auswirkung auf das Ergebnis.

6.4.2.2 Änderungen des Objektmodells und Dateiformat XML

Einleitung

Um eine kundenspezifisch erzeugte oder editierte XML-Datei über TIA Portal Openness erfolgreich in das TIA Portal zu importieren, muss die Datei definierten Schemas entsprechen.

Die XML-Dateien bestehen immer aus zwei Hauptteilen:

- Schnittstelle
- Übersetzungseinheit

Die Schemas, denen die Dateien entsprechen müssen, werden nachstehend erläutert.

Schnittstelle

Eine Schnittstelle kann mehrere Abschnitte enthalten (z. B. Input, InOut, Static): Sie finden alle diese Abschnitte im folgenden Verzeichnis:

- C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Schemas\SW.InterfaceSections_v3.xsd
- C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Schemas\SW.Interface.Snapshot .xsd

Übersetzungseinheit

Es gibt separate Schemas für die Übersetzungseinheiten der GRAPH-, KOP-/FUP-, AWL- und SCL-Bausteine. Sie finden diese Schemas in den folgenden Verzeichnissen:

- GRAPH: C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Schemas\SW.PlcBlocks.Graph_v4.xsd
- KOP/FUP: C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Schemas\SW.PlcBlocks.LADFBFD_v3.xsd
- AWL: C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Schemas\SW.PlcBlocks.STL_v3.xsd
- SCL: C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Schemas\SW.PlcBlocks.SCL_v2.xsd

Subschemas

Es gibt folgende zusätzliche Schemadefinitionen, die von allen Übersetzungseinheiten verwendet werden:

- Zugriff
- Allgemein

Zugriff

Der Zugriffsknoten beschreibt zum Beispiel:

- lokale/globale Mitglieder und konstante Nutzungen
- FB-, FC-, Anweisungsaufrufe
- DBs für Aufrufe

Sie finden das Zugriffsschema im folgenden Verzeichnis:

C:\Programme\Siemens\Automation\Portal V*\PublicAPI\V*\Schemas\SW.PlcBlocks.Access_v3.xsd

Allgemein

Hierunter fallen die allgemein verwendeten Attribute und Elemente, zum Beispiel Kommentare verschiedener Art, Texte und Token.

Sie finden das allgemeine Schema im folgenden Verzeichnis:

C:\Pogramme\Siemens\Automation\Portal V*\PublicAPI\V*.\Schemas\SW.Common_v2.xsd

Hinweis

V* bezieht sich auf die installierte Version von TIA Portal.

6.4.2.3 DBs mit Schnappschüssen exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie die DBs mit Schnappschusswerten im XML-Format exportieren, um die Werte mit verschiedenen Schnappschusszeiten zu vergleichen. Mit dem Vergleichsergebnis können sie manuell

einzelne Startwerte in der Benutzeroberfläche von TIA Portal anpassen und für eine spätere Wiederherstellung speichern.

Das Schema "SW.Interface.Snapshot.xsd" kann die exportierte XML-Datei verarbeiten. Der Snapshot Service "InterfaceSnapshot" wird im Namespace "Siemens.Engineering.SW.Blocks" bereitgestellt.

Die Dateibehandlung unterstützt den Export von DBs mit Schnappschüssen:

- Exportverzeichnis ist nicht vorhanden
- Exportverzeichnis erstellen
- Exportverzeichnis ist schreibgeschützt
- Exportverzeichnis ist bereits vorhanden

Der Snapshot Service wird für globale DBs, Instanz-DBs und Array-DBs unterstützt.

Programmcode

Ändern Sie folgenden Programmcode, um Schnappschusswerte mit dem Snapshot Service zu exportieren:

```
PlcBlock dataBlock = ...;  
InterfaceSnapshot interfaceSnapshot = dataBlock.GetService<InterfaceSnapshot>();  
interfaceSnapshot.Export(new FileInfo("C:\\temp\\MyInterfaceSnapshot.xml"),  
ExportOptions.WithReadOnly);
```

Hinweis

Der Export von Schnappschusswerten mit dem Snapshot Service ist unabhängig vom Export über die Standardschnittstelle von Openness und beeinflusst den bereits vorhandenen Export der Schnittstellenmitglieder nicht. Die exportierte XML kann nicht mehr importiert werden.

Die Schnappschusswerte werden wie folgt exportiert:

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
<Engineering version="V15 SP1" />
<DocumentInfo>
...
</DocumentInfo>
<SW.Blocks.InterfaceSnapshot ID="0">
<AttributeList>
<Name>GlobalDB</Name>
<Snapshot ReadOnly="true">
<SnapshotValues>
<Value Path="Static_1" Type="Bool">TRUE</Value>
<Value Path="Static_2[0]" Type="Int">1</Value>
<Value Path="Static_2[1]" Type="Int">2</Value>
<Value Path="Static_2[2]" Type="Int">3</Value>
<Value Path="Static_3" Type="DTL">DTL#1973-01-01-00:00:00</Value>
<Value Path="Static_4.Element_1" Type="Int">7</Value>
<Value Path="Static_4.Element_2[0]" Type="Bool">FALSE</Value>
<Value Path="Static_4.Element_2[1]" Type="Bool">TRUE</Value>
<Value Path="Static_4.Element_2[2]" Type="Bool">TRUE</Value>
<Value Path="Static_4.Element_3.Element_1" Type="Int">5</Value>
<Value Path="Static_4.Element_3.Element_2.Element_1" Type="Bool">TRUE</Value>
<Value Path="Static_4.Element_3.Element_2.Element_2[0]" Type="Int">100</Value>
<Value Path="Static_4.Element_3.Element_2.Element_2[1]" Type="Int">200</Value>
</SnapshotValues></Snapshot>
<SnapshotDate ReadOnly="true">2017-12-06T08:04:11.4590585Z</SnapshotDate>
<StructureModified ReadOnly="true">2017-12-06T08:22:13.3292585Z</StructureModified>
</AttributeList>
</SW.Blocks.InterfaceSnapshot>
</Document>
```

Wenn ein DB keine Schnappschusswerte enthält, sieht der Inhalt der exportierten Datei folgendermaßen aus:

```
<SnapshotValues xmlns="http://www.siemens.com/automation/Openness/SW/Interface/Snapshot/v1"></SnapshotValues>
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

6.4.2.4 Bausteine mit Know-how-Schutz exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist nicht online.

Einleitung

Die resultierende XML-Datei ähnelt der Exportdatei eines Bausteins ohne Knowhow-Schutz. Mit `PlcBlockProtectionProvider` können Sie einen Knowhow-geschützten Baustein entsperren, indem Sie in der Openness API das Passwort eingeben und dann einen KHP-Baustein komplett mit seinem Code exportieren. Ein Baustein kann importiert und dann wieder über die API mit einem Passwort geschützt werden.

Wenn der Baustein ohne Entsperren exportiert wird, wird nur die öffentliche Bausteinschnittstelle exportiert.

- TIA Portal-Baustein -> Entsperren -> Datei ohne Schutz exportieren
- Datei ohne Schutz importieren -> Sperren -> TIA Portal-Baustein

Die Attributliste des Bausteins zeigt an, dass der entsprechende Baustein Knowhow-Schutz besitzt.

Programmcode

Um die sichtbaren Daten eines Bausteins mit Knowhow-Schutz in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
private static void ExportBlock(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    plcBlock.Export(new FileInfo(string.Format(@"D:\Samples\{0}.xml", plcBlock.Name)),
        ExportOptions.WithDefaults);
}
```

6.4.2.5 Export/Import von SCL-Bausteinen

SCL-Anweisungen mit XML-Tags für Export

Die Exportoperation für SCL-Bausteine exportiert ihre entsprechenden XML-Tags auf der Grundlage des SCL-Anweisungstyps. Diese Operation unterstützt die SCL-Netzwerke von SCL-Anweisungen in KOP/FUP-Bausteinen von SCL-Anweisungen. Die SCL-Anweisungen sind klassifiziert als Textelemente, Operanden, Ausdrücke, Steueranweisungen usw. Das Schema SW.PlcBlocks.SCL_v2.xsd ist verfügbar, um Sie beim Verarbeiten des SCL-Bausteins mit Export-XML-Tags zu unterstützen. Die SCL-Bausteinanweisungen mit ihren entsprechenden exportierten XML-Tags und Attributen sind unten angegeben.

Neue Zeile

Neue Zeilen in SCL-Bausteinen werden durch NewLine XML-Tags dargestellt.

- Enthält das vorzeichenlose Attribut Num mit dem Standardwert 1.
- Attribut Num hat nicht den Wert 0.
- Nur bei SCL unterstützt.

SCL-Baustein	XML-Tag
	<NewLine Num="2" />

Leer

Leerzeichen in SCL-Bausteinen werden durch Blank XML-Tags dargestellt.

- Enthält das vorzeichenlose Attribut Num mit dem Standardwert 1.
- Attribut Num hat nicht den Wert 0.
- Nur bei SCL unterstützt.
- Unterstützt nicht das Attribut Integer, das in anderen Sprachen von STEP 7 verfügbar ist.

SCL-Baustein	XML-Tag
	<Blank Num="2" />

Einrückung von SCL-Bausteinanweisungen

In den Einstellungen im TIA Portal können Sie die Einrückung von SCL-Code über Options/Settings/General/Script/text editors ändern. In der folgenden Tabelle wird die Art der Einrückung auf der Grundlage des Ident-Modus definiert.

Ident-Modus	Ergebnis
Ohne	Importoperation fügt die Leerzeichen ein, wie in den Quelldateien verfügbar.
Paragraph oder Smart	Importoperation fügt die angegebenen Ident-Leerzeichen in die importierte Datei ein.

Auf der Grundlage der gewählten Einrückung wird diese in der importierten XML file des SCL-Bausteins vorgenommen.

Kommentar

Ein- und mehrzeilige Kommentare in SCL-Bausteinen werden durch LineComment XML-Tags dargestellt.

- Nur das Tag LineComment (für einsprachigen Kommentar) wird von SCL unterstützt.
- Das Tag Comment (für mehrsprachigen Kommentar) wird von SCL nicht unterstützt.
- Enthält das Attribut Inserted mit Standardwert false.
- Inserted="false" zeigt "//" einzeiligen Kommentar in SCL-Bausteinen an.
- Inserted="true" zeigt "(*)" mehrzeiligen Kommentar in SCL-Bausteinen an.
- NoClosingBracket="true" zeigt Kommentare ohne schließende Klammer in SCL-Bausteinen an. Dieses Attribut ist optional und hat den Standardwert false.
- XML zeigt keine Kommentarthierarchie in SCL-Bausteinen an.

SCL-Baustein	XML-Tag
// one line comment	<pre><LineComment> <Text>one line comment</Text> </LineComment></pre>
(* one line comment second line *)	<pre><LineComment Inserted="true"> <Text>one linecomment secondline</Text> </LineComment></pre>
(* first comment (* second comment) end first comment *)	<pre><LineComment Inserted="true"> <Text> first comment (* second comment *) end first comment</ Text> </LineComment ></pre> Der verschachtelte Kommentar ist Bestandteil des äußeren Kommentartexts.
(* comment without closing bracket	<pre><LineComment Inserted="true" NoClosingBracket="true"> <Text> comment without closing bracket</Text> </LineComment ></pre>

Region

Regionen in SCL-Bausteinen werden durch Token XML-Tags dargestellt.

- Das Text XML-Tag repräsentiert den Regionsnamen (region_name).
- Beim Attribut Text des Token XML-Tags spielt die Groß- und Kleinschreibung keine Rolle.

- Beim Importvorgang wird die Groß- und Kleinschreibung beachtet und der Editor zeigt die Schlüsselwörter wie in den TIA Portal-Einstellungen konfiguriert an.
- Wenn das Schlüsselwort `end_region` mit ";" (Semikolon) im SCL-Baustein endet, steht das Zeichen ";" im Text XML-Tag.

SCL-Baustein	XML-Tag
<pre>region myregion ... end_region here is the end of myregion</pre>	<pre><Token Text="REGION" /> <Blank /> <Text>myregion</Text> <NewLine /> ... <Token Text="END_REGION" /> <Blank /> <Text>here is the end of myregion</ Text> <NewLine /></pre>
<pre>region // here are no blanks ... end_region</pre>	<pre><Token Text="REGION" /> <NewLine /> <LineComment .../> <Token Text="END_REGION" /> <NewLine /></pre>
<pre>region ... end_region;</pre>	<pre><Token Text="REGION" /> <NewLine /> ... <Token Text="END_REGION" /> <Text>;</Text> <NewLine /></pre>

Pragma

Pragma in SCL-Bausteinen werden durch Token XML-Tags dargestellt. Die Parameter werden im XML-Tag Access mit dem Attribut Scope als LiteralConstant dargestellt.

SCL-Baustein	XML-Tag
<pre>{PRAGMA_BEGIN 'Param1', 'Param2' (*parm 2*)} // something else {PRAGMA_END}</pre>	<pre><Token Text="{ " /> <Token Text="PRAGMA_BEGIN" /> <Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantValue>'Param1'</ ConstantValue> </Constant> </Access> <Token Text=", " /> <Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantValue>'Param2'</ ConstantValue> </Constant> </Access> <Blank /> <LineComment Inserted="True"> <Text>param 2</Text> </LineComment> <Token Text=", " /> <Blank /> <Token Text="}" /> <NewLine /> <LineComment> <Text> something else</Text> </LineComment> <NewLine /> <Token Text="{ " /> <Token Text="PRAGMA_END" /> <Token Text="}" /></pre>

Konstanten: Literale Konstanten

Die Konstanten in SCL-Bausteinen werden durch Access XML-Tags dargestellt.

- Das Attribut Scope kann Werte wie LiteralConstant, TypedConstant, LocalConstant, und GlobalConstant. haben.
- Die Namen von Konstanten, denen das Zeichen "#" vorangestellt ist, werden in XML ignoriert.
- Das Zeichen "#" wird während des Importvorgangs von XML hinzugefügt.
- Der Wert globaler Konstanten, repräsentiert durch Anführungszeichen, wird in XML ignoriert.
- Die Anführungszeichen werden während des Importvorgangs von XML hinzugefügt.

Art der Konstante	SCL-Baustein	XML-Tag
Literale Konstante: Integer	#Out := 10;	<pre><Access Scope="LiteralConstant"> <Constant> <ConstantValue>10</ConstantValue> <ConstantTypeInformative="true">LINT</ConstantType> </Constant> </Access></pre>
Literale Konstante: String	#myString := 'Hello world';	<pre><Access Scope="LiteralConstant"> <Constant> <ConstantValue>Hello world</ConstantValue> <ConstantTypeInformative="true">STRING</ConstantType> </Constant> </Access></pre>
Literale Konstante: Typed	#Out := int#10;	<pre><Access Scope="TypedConstant"> <Constant> <ConstantValue>int#10</ConstantValue> </Constant> </Access></pre> Format von XML nach Export mit Einstellung ExportOptions.ReadOnly. <pre><Access Scope="TypedConstant"> <Constant> <ConstantValue>int#10</ConstantValue> <StringAttribute Name="Format" Informative="true">Dec_signed</StringAttribute> <StringAttribute Name="FormatFlags" Informative="true">TypeQualifier</StringAttribute> </Constant> </Access></pre>
Lokale Konstante	#Out := #mylocal;	<pre><Access Scope="LocalConstant"> <Constant Name="mylocal" /> </Access></pre> Format von XML nach Export mit Einstellung ExportOptions.ReadOnly <pre><Access Scope="LocalConstant"> <Constant Name="mylocal"> <ConstantType Informative="true">Int</ConstantType> <ConstantValue Informative="true">10</ConstantValue> <StringAttribute Name="Format" Informative="true">Dec_signed</StringAttribute> </Constant> </Access></pre>

Art der Konstante	SCL-Baustein	XML-Tag
Globale Konstante	#Out := "myglobal";	<pre><Access Scope="GlobalConstant"> <Constant Name="myglobal" /> </Access></pre> Format von XML nach Export mit Einstellung ExportOptions.ReadOnly. <pre><Access Scope="GlobalConstant"> <Constant Name="myglobal"> <ConstantType Informative="true">Int</ConstantType> <ConstantValue Informative="true">10</ConstantValue> <StringAttribute Name="Format" Informative="true">Dec_signed</StringAttribute> </Constant> </Access></pre>

Die Adresskonstanten werden in SCL-Bausteinen nicht unterstützt und in dieser Tabelle ignoriert.

Variablen

Die lokalen und globalen Variablen in SCL-Bausteinen werden durch Access XML-Tags dargestellt.

- Das Attribut Scope hat Werte von LocalVariable und GlobalVariable.
- Das XML-Tag für das Zuweisen des Werts 10 wird hier ignoriert.

Art der Variable	SCL-Baustein	XML-Tag
Lokale Variable	#Out := 10;	<pre><Access Scope="LocalVariable"> <Symbol> <Component Name="Out" /> </Symbol> </Access></pre>
Globale Variable	"Tag_3" := 10;	<pre><Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_3" /> </Symbol> </Access></pre> Format von XML nach Export mit Einstellung ExportOptions.ReadOnly. <pre><Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_3" /> <Address Area="Memory" Type="Int" BitOffset="96" Informative="true" /> </Symbol> </Access></pre>

Ausdrücke

Die einfachen Ausdrücke in SCL-Bausteinen werden durch Access XML-Tags dargestellt. Das Attribut Scope hat den Wert von LocalVariable für die Ausdrücke.

SCL-Baustein	XML-Tag
#a := #b + #c;	<pre> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Blank /> <Token text=":=" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="b" /> </Symbol> </Access> <Blank /> <Token text="+" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="c" /> </Symbol> </Access> <Token text=";" /> </pre>

Steuerstrukturen in SCL-Bausteinen

Die Steueranweisungen wie IF, CASE, FOR, WHILE, REPEAT, GOTO, EXIT, CONTINUE, and RETURN werden durch Token XML -Tags dargestellt.

- Die im SCL-Baustein verwendeten bedingten Symbole wie >, <, & werden in XML als Escape-Sequenzen (< > &amp) dargestellt.
- Diese Kombination von XML-Tags ist nur bei SCL-Bausteinen gültig. In anderen Sprachen wird eine Ausnahme ausgelöst.

Name des Bausteins	SCL-Baustein	XML-Tag
IF	<pre>IF #a<#c THEN ; END_IF;</pre>	<pre><Token Text="IF" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Token Text="&lt;" /> <Access Scope="LocalVariable"> <Symbol> <Component Name="c" /> </Symbol> </Access> <Blank /> <Token Text="THEN" /> <NewLine /> <Blank Num="4" /> <Token Text=";" /> <NewLine /> <Token Text="END_IF" /> <Token Text=";" /></pre>

Name des Bausteins	SCL-Baustein	XML-Tag
CASE	<pre> CASE #a OF 1 (*test*): // Statement section case 1 ; 2..4: // Statement section case 2 to 4 ; ELSE // Statement section ELSE ; END_CASE;</pre>	<pre> <Token Text="CASE" /><Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Blank /> <Token Text="OF" /> <NewLine /> <Blank Num="2"/> <Access Scope="LiteralConstant"> <Constant> <ConstantValue>1</ConstantValue> <ConstantType Informative="true">LINT</ConstantType> </Constant> </Access> <Blank /> <LineComment Inserted="true"> <Text>test</Text> </LineComment > <Token Text=":" /> <Blank /> <LineComment> <Text> Statement section case 1</Text> </LineComment > <NewLine /> <Blank Num="4"/> <Token Text=";" /> <NewLine /> <Blank Num="2"/> <Access Scope="LiteralConstant"> <Constant> <ConstantValue>2</ConstantValue> <ConstantType Informative="true">LINT</ConstantType> </Constant> </Access> <Token Text=".." /> <Blank Num="2"/> <Access Scope="LiteralConstant"> <Constant> <ConstantValue>4</ConstantValue> <ConstantType Informative="true">LINT</ConstantType> </Constant> </Access> <Blank /> <LineComment> <Text> Statement section case 2 to 4</Text> </LineComment ></pre>

6.4 Daten eines PLC-Geräts importieren/exportieren

Name des Bausteins	SCL-Baustein	XML-Tag
		<pre><NewLine /> <Blank Num="4"/> <Token Text=";" /> <NewLine /> <Blank Num="2"/> <Token Text="ELSE" /> <NewLine /> <Blank Num="4"/> <Token Text=";" /> <NewLine /> <Token Text="END_CASE" /> <Token Text=";" /></pre>

Name des Bausteins	SCL-Baustein	XML-Tag
FOR	<pre> FOR #i := #a TO #b DO // Statement section FOR ; END_FOR;</pre>	<pre> <Token Text="FOR" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="i" /> </Symbol> </Access> <Blank /> <Token Text=":=" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Blank /> <Token Text="TO" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="b" /> </Symbol> </Access> <Blank /> <Token Text="DO" /> <NewLine /> <Blank Num="2" /> <LineComment> <Text> Statement section FOR</Text> </LineComment> <NewLine /> <Blank Num="2" /> <Token Text=";" /> <NewLine /> <Token Text="END_FOR" /> <Token Text=";" /></pre>

6.4 Daten eines PLC-Geräts importieren/exportieren

Name des Bausteins	SCL-Baustein	XML-Tag
WHILE	<pre> WHILE #a<#b DO // Statement section WHILE ; END_WHILE; </pre>	<pre> <Token Text="WHILE" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Token Text="&lt;" /> <Access Scope="LocalVariable"> <Symbol> <Component Name="b" /> </Symbol> </Access> <Blank /> <Token Text="DO" /> <NewLine /> <Blank Num="2" /> <LineComment> <Text> Statement section WHILE</Text> </LineComment > <NewLine /> <Blank Num="2" /> <Token Text=";" /> <NewLine /> <Token Text="END_WHILE" /> <Token Text=";" /> </pre>

Name des Bausteins	SCL-Baustein	XML-Tag
REPEAT	<pre> REPEAT // Statement section REPEAT ; UNTIL #a<#b END_REPEAT;</pre>	<pre> <Token Text="REPEAT" /> <NewLine /> <Blank Num="2" /> <LineComment> <Text> Statement section REPEAT</Text> </LineComment > <NewLine /> <Blank Num="2" /> <Token Text=";" /> <NewLine /> <Token Text="UNTIL" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Token Text="&lt;" /> <Access Scope="LocalVariable"> <Symbol> <Component Name="b" /> </Symbol> </Access> <Blank /> <Token Text="END_REPEAT" /> <Token Text=";" /></pre>

Name des Bausteins	SCL-Baustein	XML-Tag
GOTO	<pre> here // well : // this is goto statement </pre>	XML-Beispiel für Definition von GOTO-Labels <pre> <Blank Num="3"/> <Access Scope="Label"> <Label Name="here"> <NewLine /> <Blank Num="3"/> <LineComment> <Text> well</Text> </LineComment> <NewLine /> <Token Text=":" /> <Blank /> </Label> </Access> <LineComment> <Text> this is goto statement</Text> </LineComment> </pre>
	<pre> GOTO (*comment*) here; </pre>	XML-Beispiel für Verwendung von GOTO-Labels <pre> <Token Text="GOTO" /> <Blank /> <LineComment inserted="true"> <Text>comment</Text> </LineComment> <Blank /> <Access Scope="Label"> <Label Name="here" /> </Access> <Token Text=";" /> </pre>

Referenzierende Attribute

Die referenzierenden Attribute von SCL-Bausteinen werden durch das Attribut AccessModifier des Tags Component dargestellt.

- Zum einfachen Referenzieren hat AccessModifier den Wert als Reference.
- Zur Array-Referenzierung hat AccessModifier den Wert als ReferenceToArray..

SCL-Baustein	XML-Tag
RefToUDT^(*RefToUDT*).element	<pre> <Symbol> <Component Name="RefToUDT" AccessModifier="Reference" /> <Token Text="^" /> <LineComment Inserted="True"> <Text>RefToUDT</Text> </LineComment> <Token Text="." /> <Component Name="element" /> </Symbol> </pre>
RefToArrayOfUDT^(*RefToArrayOfUDT*)[#i].element	<pre> <Symbol> <Component Name="RefToArrayOfUDT" AccessModifier="ReferenceToArray" /> <Token Text="^" /> <LineComment Inserted="True"> <Text>RefToArrayOfUDT</Text> </LineComment> <Token Text="[" /> <Access Scope=LocalVariable> <Symbol> <Component Name="i" /> </Symbol> </Access> <Token Text="]" /> </Component> <Token Text="." /> <Component Name="element" /> </Symbol> </pre>

6.4.2.6 Export/Import strukturierter Typen von SCL-Bausteinen

Strukturierte SCL-Typen mit XML-Tags für den Export

In die strukturierten SCL-Typen können Sie Leerzeichen, neue Zeilen und Kommentare zu den SCL-Anweisungen einfügen. Die strukturierten SCL-Anweisungen mit ihren entsprechenden exportierten XML-Tags und Attributen sind unten angegeben.

Globaler Zugriff

In SCL-Anweisungen stehen Variablen und Konstanten für globalen Zugriff in Anführungszeichen. Die Kommentare zwischen den Variablen und Adressteilen werden durch das LineComment XML-Tag dargestellt.

SCL-Baustein	XML-Tag
"Datenbaustein_1".(*Kommentare 1*)Statisch_1(*Kommentare 2*).Statisch_2	<pre> <Access Scope="GlobalVariable"> <Symbol> <Component Name="Data_block_1" /> <Token Text="." /> <LineComment Inserted="True"> <Text>comment 1</Text> </LineComment> <Component Name="Static_1" /> <LineComment Inserted="True"> <Text>comment 2</Text> </LineComment> <Token Text="." /> <Component Name="Static_2" /> </Symbol> </Access> </pre>
"Data_block_1".Static_1 := 10	Format von XML nach Export mit Einstellung ExportOptions.None <pre> <Access Scope="GlobalVariable"> <Symbol> <Component Name="Data_block_1" /> <Token Text="." /> <Component Name="Static_1" /> </Symbol> </Access> </pre> Format von XML nach Export mit Einstellung ExportOptions.ReadOnly <pre> <Access Scope="GlobalVariable"> <Symbol> <Component Name="Data_block_1" /> <Token Text="." /> <Component Name="Static_1" /> <Address Area="DB" Type="Word" BlockNumber="1" BitOffset="0" Informative="true" /> </Symbol> </Access> </pre>

Verwendung von Anführungszeichen und

Die in der ersten Ebene verwendeten Anführungszeichen beschreiben die Art der Variable und dienen als Escape-Sequenzen für Sonderzeichen in SCL-Anweisungen. Wenn Anführungszeichen in der ersten Ebene verwendet werden, definieren sie die Variable als globale Variable. Wenn die Anführungszeichen nach # verwendet werden, stellen sie die Escape-Sequenz von Sonderzeichen wie # und von Leerzeichen dar.

- Zum Darstellen der unterschiedlichen Verwendung dient in der XML-Datei das Tag `BooleanAttributes` mit dem Attribut `Name`. Der Name enthält Werte wie `HasQuotes` und `HasHash`.
- Um im `Scope`-Attribut die Struktur zu definieren, wird # festgelegt.
- Diese Werte gelten nur für SCL.
- Die Standardwerte für diese Tags sind anfänglich `FALSE`, aber die Werte werden auch mit der Einstellung `ExportOptions.WithDefaults` nie exportiert.

SCL-Baustein	XML-Tag
"a".#b."c".#"d"	<pre> <Access Scope="GlobalVariable"> <Symbol> <Component Name="a" /> <Token Text="." /> <Component Name="b"> <BooleanAttribute Name="HasHash">TRUE</ BooleanAttribute> </Component> <Token Text="." /> <Component Name="c"> <BooleanAttribute Name="HasQuotes">TRUE</ BooleanAttribute> </Component> <Token Text="." /> <Component Name="d"> <BooleanAttribute Name="HasQuotes">TRUE</ BooleanAttribute> <BooleanAttribute Name="HasHash">TRUE</ BooleanAttribute> </Component> </Symbol> </Access /> </pre>

Array

SCL erlaubt das Einfügen von Kommentaren in die Arrayindizes um "[" und "]". Zum Kennzeichnen des Vorhandenseins eines Arrays dient in der XML-Datei das Attribut AccessModifier im Tag Component.

- Wenn Accessmodifier den Wert Array enthält, ist ein untergeordnetes Tag Access obligatorisch, um die Indexvariable des Arrays anzuzeigen.
- Der Standardwert für AccessModifier ist None.

SCL-Baustein	XML-Tag
#a.b[#i+#j,#k+#l].c	<pre> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> <Token Text="." /> <Component Name="b" AccessModifier="Array" /> <Token Text="[" /> <Access Scope=LocalVariable> <Symbol> <Component Name="i" /> </Symbol> </Access> <Token Text="+" /> <Access Scope=LocalVariable> <Symbol> <Component Name="j" /> </Symbol> </Access> <Token Text="," /> <Access Scope=LocalVariable> <Symbol> <Component Name="k" /> </Symbol> </Access> <Token Text="+" /> <Access Scope=LocalVariable> <Symbol> <Component Name="l" /> </Symbol> </Access> <Token Text="]" /> </Component> <Token Text="." /> <Component Name="c" /> </Symbol> </Access> </pre>

Absoluter Zugriff

SCL ermöglicht verschiedene Zugriffsarten wie Absolut, Absolut-Offset, Gemischt (Datenbank und Membervariable), Slice, Peripher und Direkt. Der absolute Zugriff wird in XML durch das Tag Address dargestellt.

- Das Zeichen % in der DB wird in XML nicht geschrieben. Es wird beim Import automatisch erzeugt.
- Zwischen den Teilen der Adresse sind Leerzeichen zulässig.

SCL-Baustein	XML-Tag
%DB20 . DBW10	<pre><Access Scope="Address"> <Symbol> <Address Area="DB" BlockNumber="20" /> <Blank /> <Token Text="." /> <Blank /> <Address Area="DB" BitOffset="80" Type="Word"/> </Symbol> </Access></pre>
%DB20.DBX10.3 := true;	Folgende XML gilt für alle Sprachen außer SCL. <pre><Access Scope="Address"> <Address Area="DB" BlockNumber="20" BitOffset="83" Type="Bool" /> </Access></pre> Folgende XML gilt für SCL. <pre><Access Scope="Address"> <Symbol> <Address Area="DB" BlockNumber="20" /> <Token Text="." /> <Address Area="DB" BitOffset="83" Type="Bool"/> </Symbol> </Access></pre>

Zugriffsart "Absolut-Offset"

In AWL stellt das Tag AbsoluteOffset die Zugriffsart "Absolut-Offset" dar. In SCL wird das Tag Address für den absoluten Zugriff verwendet.

SCL-Baustein	XML-Tag
#Input_DB_ANY.%DBX2.3 := TRUE;	<pre><Access Scope="LocalVariable"> <Symbol> <Component Name="Input_DB_ANY" /> <Token Name="." /> <Address BitOffset="19" Type="Bool" /> </Symbol> </Access></pre>

Zugriffsart "Slice"

In SCL wird das Attribut SliceAccessModifier nicht unterstützt, und die Zugriffsart "Slice" wird durch das Tag Token dargestellt.

SCL-Baustein	XML-Tag
"tag_1"(*1*) . (*2*)member(*3*) . (*4*) %x1	<pre><Access Scope="GlobalVariable"> <Symbol> <Component Name="tag_1" /> <LineComment Inserted="True"> <Text>1</Text> </LineComment> <Token Text="." /> <LineComment Inserted="True"> <Text>2</Text> </LineComment> <Component Name="member"/> <LineComment Inserted="True"> <Text>3</Text> </LineComment> <Token Text="." /> <LineComment Inserted="True"> <Text>4</Text> </LineComment> <Token Text="%x1" /> </Symbol> </Access></pre>

Peripherer Zugriff

Der periphere Zugriff wird durch das Tag Token dargestellt.

SCL-Baustein	XML-Tag
"tag_1" (*1*) . (*2*) member:P	<pre> <Access Scope="GlobalVariable"> <Symbol> <Component Name="tag_1" /> <LineComment Inserted="True"> <Text>1</Text> </LineComment> <Token Text="." /> <LineComment Inserted="True"> <Text>2</Text> </LineComment> <Component Name="member"/> <Token Text=":P" /> </Symbol> </Access> </pre>

Zugriffsart "Direkt"

Die Anweisungen TypeOf und TypeOfDB werden entweder mit systemdefiniertem oder benutzerdefiniertem Typ ausgeführt. Angegeben sind die Typen im Tag Access mit dem Attribut Scope und den Werten SystemType und UserType.

SCL-Baustein	XML-Tag
Beispiel für systemdefinierten Typ if TypeOf(#inVariant) = TO_SpeedAxis then ... end_if	<pre> <Token text="=" /> </Blank> <Access Scope="SystemType"> <DataType>TO_SpeedAxis</DataType> </Access> </pre>
Beispiel für benutzerdefinierten Typ if TypeOf(#inVariant) = "aUserDefinedType" then ... end_if	<pre> <Token text="=" /> </Blank> <Access Scope="UserType"> <DataType>aUserDefinedType</ DataType> </Access> </pre>

6.4.2.7 Export/Import von SCL-Aufrufbausteinen

SCL-Aufrufbausteine mit XML-Tags für den Export

SCL-Aufrufparameter werden in XML durch das Tag Parameter dargestellt. Das Attribut informative dient zum Darstellen der nicht zugewiesenen Parameter und Rückgabewerte wie Zeitstempel, Merkerinformationen usw. Das XML-Format folgt der gleichen willkürlichen Reihenfolge wie im SCL-Baustein.

Nachstehend ein Beispiel für einen Bausteinaufruf:

SCL-Baustein	XML-Tag
#Callee_Instance(Input_1 := 5);	Format von XML nach Export mit Einstellung ExportOptions.None <pre> <Access Scope="Call"> <CallInfo BlockType="FB"> <Instance Scope="LocalVariable"> <Component Name="Callee_Instance" /> </Instance> <Token text="(" /> <Parameter Name="Input_1"> <Blank /> <Token text=":=" /> <Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Int</ ConstantType> <ConstantValue>5</ ConstantValue> </Constant> </Access> </Parameter> <Token text=")" /> </CallInfo> </Access> <Token text=";" /> </pre>
	Format von XML nach Export mit Einstellung ExportOptions.ReadOnly <pre> <Access Scope="Call"> <CallInfo BlockType="FB"> <IntegerAttribute Name="BlockNumber" Informative="true">1</ IntegerAttribute> <DateAttribute Name="ParameterModifiedTS" Informative="true">2016-10-24T08:27: 34</DateAttribute> <Instance Scope="LocalVariable"> <Component Name="Callee_Instance" /> </Instance> <Token text="(" /> <Parameter Name="Input_1"> <StringAttribute Name="InterfaceFlags" Informative="true">S7_Visible</ StringAttribute> <Blank /> <Token text=":=" /> </pre>

SCL-Baustein	XML-Tag
	<pre><Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Int</ ConstantType> <ConstantValue>5</ ConstantValue> </Constant> </Access> </Parameter> <Token text=") " /> </CallInfo> </Access> <Token text=";" /></pre>

Beispiel für unverbundene Parameter

Der FB hat vier Parameter, wobei a, b, c sowie d. b und d nicht verbunden sind.

SCL-Baustein	XML-Tag
"Block_4_DB" (a:=TRUE,c:=TRUE) ;	<pre> <Access Scope="Call"> <CallInfo Name="Block_4" BlockType="FB"> <Instance Scope="GlobalVariable"> <Component Name="Block_4_DB" /> </Instance> <Token text="(" /> <Parameter Name="a"> <Token text=":=" /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Bool</ ConstantType> <ConstantValue>TRUE</ ConstantValue> </Constant> </Access> </Parameter> <Token text="," /> <Parameter Name="b" Informative="true"/> <Parameter Name="c" > <Token text=":=" /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Bool</ ConstantType> <ConstantValue>True</ ConstantValue> </Constant> </Access> </Parameter> <Parameter Name="d" Informative="true"/> <Token text=")" /> </CallInfo> </Access> </pre>

Beispiel für "ein Parameter"

Der SCL-Baustein ermöglicht Ihnen das Weglassen des Parameternamens. Dieser Parameter wird durch das Tag NamelessParameter dargestellt. Das Tag NamelessParameter hat keine Attribute und gilt nur für SCL.

SCL-Baustein	XML-Tag
"Block_4_DB" (TRUE) ;	<pre><Access Scope="Call"> <CallInfo Name="Block_4" BlockType="FB"> <Instance Scope="GlobalVariable"> <Component Name="Block_4_DB" /> </Instance> <Token text="(" /> <NamelessParameter> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Bool</ ConstantType> <ConstantValue>TRUE</ ConstantValue> </Constant> </Access> </NamelessParameter> <Token text=")" /> </CallInfo> </Access></pre>

Ausdruck als Aktualparameter

SCL-Baustein	XML-Tag
#Callee_Instance(Input_1 := #a+3);	<pre> <Access Scope="Call"> <CallInfo BlockType="FB"> <Instance Scope="LocalVariable"> <Component Name="Callee_Instance" /> </Instance> <Token text="(" /> <Parameter Name="Input_1"> <Blank /> <Token text=":=" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Token text="+" /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Int</ ConstantType> <ConstantValue>3</ ConstantValue> </Constant> </Access> </Parameter> <Token text=")" /> </CallInfo> </Access> <Token text=";" /> </pre>

Ausdruck als Aktualparameter ohne Formalparameter

SCL-Baustein	XML-Tag
#Callee_Instance(#a+3);	<pre> <Access Scope="Call"> <CallInfo BlockType="FB"> <Instance Scope="LocalVariable"> <Component Name="Callee_Instance" /> </Instance> <Token text="(" /> <NamelessParameter> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Token text="+" /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Int</ ConstantType> <ConstantValue>3</ ConstantValue> </Constant> </Access> </NamelessParameter> <Token text=")" /> </CallInfo> </Access> <Token text=";" /> </pre>

Funktionsaufruf

SCL-Baustein	XML-Tag
#myInt := "MyFunction"(Param_1 := 1, Param_2 := 15, Param_3 := TRUE);	<pre> <Access Scope="LocalVariable"> <Symbol> <Component Name="myInt" /> </Symbol> </Access> <Blank /> <Token text=":=" /> <Blank /> <Access Scope="Call"> <CallInfo Name="MyFunction" BlockType="FC"> <Token text="(" /> <Parameter Name="Param_1"> ... </pre>

Absoluter Aufruf

In SCL kann der Aufruf über die absolute Adresse des DB initiiert werden. Wegen der absoluten Adresse ist das Attribut Name des Knotens CallInfo leer.

Eine wiederherstellbare Ausnahme wird durch den Import ausgelöst, wenn

- Ein Knoten "Address" mit gültigem Wert des Attributs Name verfügbar ist.
- Der Knoten "Address" nicht vorhanden ist und kein gültiger Wert des Attributs Name vorhanden ist.

SCL-Baustein	XML-Tag
%DB20 (. . .) ;	<pre> <Access Scope="Call"> <CallInfo Name="" BlockType="FB"> <Instance Scope="GlobalVariable"> <Address Area="DB" BlockNumber="20" /> </Instance> <Token text="(" /> <Parameter> ... </Parameter> <Token text=")" /> </CallInfo> </Access> </pre>

Anweisung

Die Anweisung im SCL-Baustein wird während des Importvorgangs in der Systembibliothek geprüft, und die Anweisungsversionen werden beim Exportvorgang nicht exportiert.

Der allgemeine Anweisungstyp ist unten angegeben.

SCL-Baustein	XML-Tag
<pre>#myInt := ATTACH(OB_NR := 1, EVENT := 15, ADD := TRUE);</pre>	Format von XML nach Export mit Einstellung ExportOptions.ReadOnly <pre><Access Scope="LocalVariable"> <Symbol> <Component Name="myInt" /> </Symbol> </Access> <Blank /> <Token text=":=" /> <Blank /> <Access Scope="Call"> <Instruction Name="ATTACH"> <Token text="(" /> <Parameter Name="OB_NR"> <Blank /> <Token text=":=" /> <Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>OB_ATT</ ConstantType> <ConstantValue>1</ ConstantValue> </Constant> </Access> </Parameter> <Token text="," /> <Blank /> <Parameter Name="EVENT"> <Blank /> <Token text=":=" /> <Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>EVENT_ATT</ ConstantType> <ConstantValue>15</ ConstantValue> </Constant> </Access> </Parameter> <Token text="," /> <Blank /> <Parameter Name="ADD"> <Blank /> <Token text=":=" /> <Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Bool</ ConstantType> <ConstantValue>TRUE</ ConstantValue> </Constant> </Access> </Parameter> </Instruction> </Access></pre>

SCL-Baustein	XML-Tag
	<pre></Constant> </Access> </Parameter> <Parameter Name="RET_VAL" Informative="true" /> <Token text=")" /> </Instruction> </Access> <Token text=";" /></pre>

Anweisung mit Vorlage

Wenn der Vorlagenparameter den Anweisungsnamen ergänzt, ist der Export des Vorlagenparameters notwendig. Wenn ein "TemplateValue" -Tag mit Attribut Type="Type" auf das Instruction-Tag folgt, verkettet der Importvorgang den Vorlagenwert mit dem Anweisungsnamen.

SCL-Baustein	XML-Tag
<pre>"tag_4" := MIN_DINT(IN1:="Tag_1", IN2:="Tag_2", IN3:="Tag_3");</pre>	<pre><Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_4" /> </Symbol> </Access> ... <Access Scope="Call"> <Instruction Name="MIN"> <TemplateValue Name="value_type" Type="Type">DInt</ TemplateValue> ... <Parameter Name="IN1"> ... <Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_1" /> </Symbol> </Access> </Parameter> ... <Parameter Name="IN2">... <Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_2" /> </Symbol> </Access> </Parameter> ... <Parameter Name="IN3"> ... <Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_3" /> </Symbol> </Access> </Parameter> ... </Instruction> </Access> ...</pre>

Umwandlung

Bei Umwandlungsfunktionen werden der reale Anweisungsname und seine Vorlage nicht exportiert. Stattdessen wird der im SCL-Baustein verwendete Name exportiert.

SCL-Baustein	XML-Tag
<pre>#output_1 := TIME_TO_S5TIME(#input_1);</pre>	<pre><Access Scope="LocalVariable"> <Symbol> <Component Name="output_1" /> </Symbol> </Access> ... <Access Scope="Call"> <Instruction Name="TIME_TO_S5TIME"> <Token text="(" /> <NamelessParameter> <Access Scope="LocalVariable"> <Symbol> <Component Name="input_1" /> </Symbol> </Access> </NamelessParameter> <Token text=")" /> </Instruction> </Access> ...</pre>

Anweisung mit Instanz

Instanz und Anweisung werden durch Leerzeichen getrennt. Leerzeichen sind optional, und sie können durch neue Zeilen und Kommentare dargestellt werden. Die Anweisung TON wird durch das Attribut Name des Tags Instruction dargestellt.

SCL-Baustein	XML-Tag
IEC_Timer_0_DB . TON (IN:="Tag_1", PT:="Tag_2");	<pre> <Access Scope="GlobalAccess"> <Symbol> <Component Name="IEC_Timer_0_DB" /> </Symbol > </Access> <Blank /> <Token text="." /> <Blank /> <Access Scope="Call"> <Instruction Name="TON"> <Blank /> <Token text="(" /> <Parameter Name="IN"> <Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_1" /> </Symbol> </Access> </Parameter> ... <Token text=")" /> </Instruction> </Access> <Token text=";" /> </pre>

Alarmkonstante

Die Alarmkonstanten werden nur in S7 400-PLCs verwendet, und der exportierte XML-Code hat Ähnlichkeit mit anderen Sprachen.

SCL-Baustein	XML-Tag
"Block_1_DB"(16#0000_0001);	Format von XML nach Export mit Einstellung ExportOptions.None <pre> <Access Scope="Call"> <CallInfo Name="Block_1" BlockType="FB"> <Instance Scope="GlobalVariable"> <Component Name="Block_1_DB" /> </Instance> <NamelessParameter> <Access Scope="AlarmConstant"> <Constant> <ConstantType>C_Alarm_8</ ConstantType> <ConstantValue>16#0000_0001</ ConstantValue> </Constant> </Access> </NamelessParameter > </CallInfo> </Access> </pre> Format von XML nach Export mit Einstellung ExportOptions.ReadOnly <pre> <Access Scope="Call"> <CallInfo Name="Block_1" BlockType="FB"> <Instance Scope="GlobalVariable"> <Component Name="Block_1_DB" /> </Instance> <NamelessParameter> <Access Scope="AlarmConstant" > <Constant> <ConstantValue>16#00000001</ ConstantValue> <ConstantType>C_Alarm</ ConstantType> <StringAttribute Name="Format" Informative="true">Hex</ StringAttribute> </Constant> </Access> </NamelessParameter> </CallInfo> </Access> </pre>

ENO (Freigabeausgang)

Um den Freigabeausgang ENO im SCL-Baustein zu unterstützen, wird im Tag "Access" das Attribut "Scope" mit dem Wert "PredefinedVariable" verwendet. Es enthält auch das Tag "PredefinedVariable" als untergeordnetes Element des Tags Access.

- Das Tag "PredefinedVariable" hat ein obligatorisches Attribut "Name".
- Der Umfang "PredefinedVariable" und das Tag "PredefinedVariable" sind nur für SCL zulässig.

SCL-Baustein	XML-Tag
Call (... , ENO => ENO);	<pre> <Access Scope="Call"> <CallInfo BlockType="FC"> <Token text="(" /> ... <Token text="," /> <Blank /> <Parameter Name="ENO"> <Blank /> <Token text="=>" /> <Blank /> <Access Scope="PredefinedVariable"> <PredefinedVariable Name="ENO" /> </Access> </Parameter> <Token text=")" /> </CallInfo> </Access> <Token text=";" /> </pre>
IF ENO = #c THEN ...	<pre> <Token text="IF" /> <Blank /> <Access Scope="PredefinedVariable"> <PredefinedVariable Name="ENO" /> </Access> <Blank /> <Token Text="=" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="c" /> </Symbol> </Access> <Blank /> <Token Text="THEN" /> </pre>

6.4.2.8 Mehrsprachige Kommentare in SCL exportieren/importieren

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

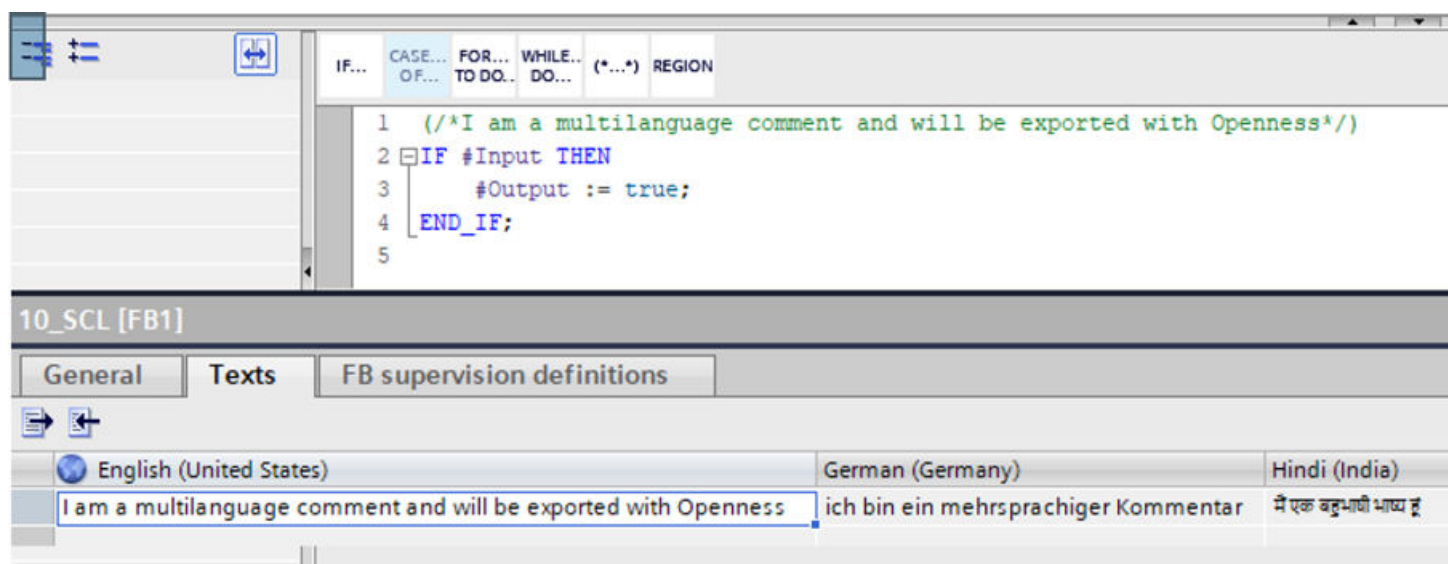
Anwendung

In TIA Portal Openness können mehrsprachige Kommentare im SCL-Editor auf eine der folgenden Arten importiert und exportiert werden:

- Beim Export von SCL in Openness werden alle mehrsprachigen Kommentare mit Übersetzung entsprechend den aktivierten Projektsprachen in die gleiche Openness xml-Datei exportiert.
- Beim Import von Openness in SCL werden alle mehrsprachigen Kommentare mit Übersetzung entsprechend den aktivierten Projektsprachen aus der Openness xml-Datei zurückimportiert. Dies muss im Textbereich für das Projekt angezeigt werden. Die Sprache wird aktiviert, wenn sie während des Imports im Projekt nicht aktiviert war.
- Nach dem Import ist zu prüfen, ob alle mehrsprachigen Kommentare im SCL-Editor mit den entsprechenden Kommentaren im Projekttext verknüpft sind.

Beispiel

Die folgende Abbildung zeigt ein Beispiel eines mehrsprachigen Kommentars im SCL-Editor:



Die folgende Abbildung zeigt ein Beispiel eines Exports eines mehrsprachigen Kommentars in die Openness XML-Datei:

```
<Comment Inserted="true" UId="21">  
  <MultilanguageText Lang="en-US">I am a multilanguage comment and will be exported with Openness</MultilanguageText>  
  <MultilanguageText Lang="de-DE">ich bin ein mehrsprachiger Kommentar</MultilanguageText>  
  <MultilanguageText Lang="hi-IN">मैं एक बहुभाषी भाष्य हूँ</MultilanguageText>  
</Comment>
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.4.2.9 F-Bausteine exportieren

Fehlersichere Bausteine exportieren

Es ist jetzt möglich, die fehlersicheren Bausteine zu importieren und zu exportieren, doch F-Systembausteine können nicht exportiert werden.

Fehlersichere Bausteine importieren

Konsistente F-Bausteine können exportiert und wieder importiert werden. Sie werden als F-Bausteine erstellt.

Sie werden als Standardbausteine importiert, wenn Sie das Präfix "F_" aus dem Wert sämtlicher Attribute "ProgrammingLanguage" entfernen.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Bausteine exportieren (Seite 1395)

Bausteine mit Know-how-Schutz exportieren (Seite 1352)

6.4.2.10 System-Bausteine exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Das Projekt enthält einen Systembaustein.

6.4 Daten eines PLC-Geräts importieren/exportieren

- Der Systembaustein ist kein F-Baustein.
- Die PLC ist nicht online.

Einleitung

In der Baustein-Zusammensetzung sind nur sichtbare Systembausteine verfügbar. Beispielsweise sind darin keine FB- oder FC-Bausteine verfügbar. Die resultierende XML-Datei ähnelt der Exportdatei eines Bausteins.

Programmcode

Um die sichtbaren Daten eines Bausteins in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports system blocks
private static void ExportSystemBlocks(PlcSoftware plcsoftware)
{
    PlcSystemBlockGroup sbSystemGroup = plcsoftware.BlockGroup.SystemBlockGroups[0];
    foreach (PlcSystemBlockGroup group in sbSystemGroup.Groups)
    {
        foreach (PlcBlock block in group.Blocks)
        {
            block.Export(new FileInfo(string.Format(@"D:\Samples\{ 0 }.xml", block.Name)),
                ExportOptions.WithDefaults);
        }
    }
}
```

6.4.2.11 GRAPH-Bausteine mit mehrsprachigem Text exportieren

XML-Struktur von GRAPH-Bausteinen mit mehrsprachigem Text

Die Export-XML von GRAPH-Bausteinen enthält die übersetzten Schrittnamen und Transitionsnamen des Graphen. Diese übersetzten mehrsprachigen Texte werden jeweils unter dem übergeordneten Element Step und Transition als Elemente StepName und TransitionName dargestellt. Diese Elemente enthalten für jede unterstützte Sprache ein Element MultiLanguageText. Die Texte für die Sprachen, die nicht explizit eingestellt sind, werden nicht exportiert. Wenn keine Übersetzung vorhanden ist, werden die Elemente StepName und TransitionName nicht exportiert. Die Elemente StepName und TransitionName sind optional. Die TIA Portal Openness XML-Importoperation löst bei den Graph-Versionen kleiner als V5.0 eine wiederherstellbare Ausnahme aus.

Beispiel für das Element StepName

```
<Steps>
  <Step Number="1" Init="true" Name="Step1" MaximumStepTime="T#10S" WarningTime="T#7S">
    <StepName>
      <MultiLanguageText Lang="de-DE">stepDE</MultiLanguageText>
      <MultiLanguageText Lang="en-US">stepEN</MultiLanguageText>
      <MultiLanguageText Lang="it-CH">stepIT</MultiLanguageText>
    </StepName>
    ..
  </Step>
  ..
</Steps>
```

Beispiel für das Element TransitionName

```
<Transitions>
  <Transition IsMissing="false" Name="Trans1" Number="1" ProgrammingLanguage="LAD">
    <TransitionName>
      <MultiLanguageText Lang="de-DE">transDE</MultiLanguageText>
      <MultiLanguageText Lang="en-US">transEN</MultiLanguageText>
      <MultiLanguageText Lang="it-CH">transIT</MultiLanguageText>
    </TransitionName>
    ..
  </Transition>
  ..
</Transitions>
```

6.4.2.12 Baustein importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Einleitung

Die TIA Portal Openness API unterstützt den Import von Bausteinen mit den Programmiersprachen "AWL", "FUP", "GRAPH", "SCL" oder "KOP" aus einer XML-Datei. Die folgenden Bausteintypen werden unterstützt:

- Funktionsbausteine (FB)
- Funktionen (FC)
- Organisationsbausteine (OB)
- Globale Datenbausteine (DB)

Hinweis

Importieren optimierter Datenbausteine

Optimierte Datenbausteine werden nur von CPUs ab S7-1200 unterstützt. Wenn Sie optimierte Datenbausteine in S7-300 oder S7-400 importieren, wird eine Ausnahme ausgelöst und der Import schlägt fehl.

Reaktion auf den Import

Beim Importieren eines Bausteins gelten die folgenden Regeln:

- Die XML-Datei kann weniger Daten enthalten als der Baustein im Projekt, z. B. weniger Parameter.
- Redundante Informationen wie Aufrufinformationen müssen im Projekt und in der XML-Datei identisch sein. Andernfalls wird eine Ausnahme ausgelöst.
- Die Daten in der XML-Datei dürfen im Hinblick auf ihre Übersetzungsfähigkeit im TIA Portal „inkonsistent“ sein.
- Attribute mit den Attributen "ReadOnly=True" und "Informative=True" werden nicht importiert.
- Fehlende Instanz-DBs werden nicht automatisch erstellt.
- Wenn in der XML-Datei keine Bausteinnummer angegeben ist, wird die Bausteinnummer automatisch zugeordnet.
- Wenn der Baustein im Projekt nicht vorhanden ist und in der XML-Datei keine Versionsinformationen angegeben sind, wird die Version "0.1" zugeordnet.

Programmcode

Ändern Sie folgenden Programmcode:

```
//Import blocks
private static void ImportBlocks(PlcSoftware plcSoftware)
{
    PlcBlockGroup blockGroup = plcSoftware.BlockGroup;
    IList<PlcBlock> blocks = blockGroup.Blocks.Import(new FileInfo(@"D:\Blocks\myBlock.xml"),
    ImportOptions.Override);
}
```

Ändern Sie folgenden Programmcode:

```
//Import system blocks
private static void ImportSystemBlocks(PlcSoftware plcSoftware)
{
    PlcBlockSystemGroup systemblockGroup =
    plcSoftware.BlockGroup.SystemBlockGroups[0].Groups[0];
    IList<PlcBlock> blocks = systemblockGroup.Blocks.Import(new
    FileInfo(@"D:\Blocks\myBlock.xml"), ImportOptions.Override);
}
```

6.4.2.13 Bausteine exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Einleitung

Die API-Schnittstelle unterstützt das Exportieren konsistenter Bausteine und Anwenderdatentypen in eine XML-Datei.

Die XML-Datei erhält den Namen des Bausteins. Die folgenden Bausteintypen werden unterstützt:

- Funktionsbausteine (FB)
- Funktionen (FC)
- Organisationsbausteine (OB)
- Globale Datenbausteine (DB)

Die folgenden Programmiersprachen werden unterstützt:

- AWL
- FUP
- KOP
- GRAPH
- SCL

Für alle Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` in alle Bausteine exportiert (siehe Export von Projektierungsdaten (Seite 1253)). In Fettdruck dargestellten Attribute werden immer exportiert.

Weitere Informationen finden Sie im Informationssystem des TIA Portals unter „Übersicht über die Bausteinattribute“.

Attribut	Typ	Standardwert	Schreibgeschützt
AutoNumber	Bool	true	false
CodeModifiedDate	DateTime	-	true
CompileDate	DateTime	-	true
CreationDate	DateTime	-	true
HeaderAuthor	String	""	false
HeaderFamily	String	""	false
HeaderName	String	""	false
HeaderVersion	String	„0,1“	false
Interface	String	leere Schnittstelle	false
InterfaceModifiedDate	DateTime	-	true
IsConsistent	Bool	-	true
IsKnowHowProtected ¹	Bool	false	true
IsWriteProtected	Bool	false	true
MemoryLayout	enum MemoryLayout	-	false
ModifiedDate	DateTime	-	true
Name	String	-	false
Number	Int32	nächste verfügbare Nummer	false
ParameterModified	DateTime	-	true
PLCSimAdvancedSupport	Bool	false	true
ProgrammingLanguage	enum ProgrammingLanguage	-	false
StructureModified	DateTime	-	true

¹ Das Attribut IsKnowHowProtected gilt auch für UDT.

Hinweis

Es gibt bestimmte Bedingungen, unter denen das Attribut MemoryLayout schreibgeschützt ist.

Dynamisch zugängliche allgemeine Attribute

Die folgenden Attribute sind nur unter bestimmten Bedingungen schreibgeschützt:

Attribut	Zustand ReadOnly
AutoNumber	All KnowHowProtected blocks, All Classic OBs; Plus OBs: DiagnosticErrorInterrupt, IOAccessError, ProgrammingError, PullOrPlugOfModules, RackOrStationFailure, Status, TimeErrorInterrupt, Update
HeaderVersion	System- and KnowHowProtected DBs; ArrayDBs that originated from system library
HeaderName	
HeaderFamily	
HeaderAuthor	
Memory-Layout	Classic blocks, System Know how protected blocks, ArrayDBs, IDBoFBs, Graph blocks

Für ArrayDB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für ArrayDB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
ArrayDataType	String	-	true
ArrayLimitUpperBound	Int32	-	true

Für DB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für DB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
IsOnlyStoredInLoadMemory	Bool	false	false
IsPLCDB	Bool	false	false
IsWriteProtectedInAS	Bool	false	false

Für FB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für FB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
AssignedProDiagFB	String	-	-
ISMultInstanceCapable	Bool	-	true
Supervisions	String	no supervisions	true bei IDB of FB und false bei FB

Für DB- und FB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für DB- und FB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
IsIECCheckEnabled	Bool	false	false
IsRetainMemResEnabled ¹	Bool	false	false
MemoryReserve	Unsigned	0	false
RetainMemoryReserve ²	Unsigned	0	false

² Wenn der Wert des Attributs "IsRetainMemResEnabled" "false" ist und das Attribut "RetainMemoryReserve" ungleich "0" ist, wird eine Ausnahme ausgelöst.

Für FB-, DB- und IDB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für FB-, DB- und IDB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
DownloadWithoutReinit	Bool	false	true

Für FB- und FC-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für FB- und FC-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
LibraryType	String	-	true
LibraryTypeVersionGuid	String	-	true

Für FB- und FC-Bausteine (AWL) geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für FB- und FC-Bausteine (AWL) exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
ParameterPassing	Bool	false	false

Für FB, FC und Instanz-DB von FB-Bausteinen geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für FB, FC und Instanz-DB von FB-Bausteinen exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
UDABlockProperties	String	""	false
UDAEnableTagReadback	Bool	false	false

Für Instanz-DBs von FBs und UDTs geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für Instanz-DBs von FB- und UDT-Bausteinen exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
InstanceOfName	String	""	false
InstanceOfNumber	Unsigned Short	-	true
InstanceOfType	enum BlockType	-	true
OfSystemLibElement	String	""	false
OfSystemLibVersion	String	""	false

Für OB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für OB-Bausteine für bestimmte Plus-PLCs exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
ApplicationCycle	Single	-	true
AutomaticMinimum	Bool	-	true
ConstantName	String	-	true
CycleTimeDistributedIO	Single	-	true
CyclicApplicationCycleTime	Single	-	true
CyclicTime	Int32	100000	true
DataExchangeMode	OBDataExchangeMode	Cyclic	true
DelayTime	Double	-	true
DistributedIOName	String	-	true
DistributedIONumber	Int32	-	true

6.4 Daten eines PLC-Geräts importieren/exportieren

Attribut	Typ	Standardwert	Schreibgeschützt
EnableTimeError	Bool	-	true
EventClass	String	-	true
EventsToBeQueued	Int32	-	true
EventThresholdForTimeError	Int32	-	true
Execution	OBExecution	Never	true
Factor	Single	-	true
PhaseOffset	Int32	0	true
PriorityNumber	Int32	-	true
ProcessImagePartNumber	UInt32	-	true
ReportEvents	Bool	-	true
SecondaryType ³	String	-	false
StartDate	DateTime	1/1/2012	true
SynchronousApplicationCycleTime	Single	-	true
TimeMode	OBTimeMode	System	true
TimeOfDay	DateTime	12:00 AM	true
TransformationDBNumber	UInt16	0xffff	true

³ Beim Exportieren eines OB wird anhand der OB-Nummer zusätzlich der SecondaryType festgelegt. Die Zuordnung wird während des Importvorgangs geprüft. Ist die Zuordnung falsch, wird eine Ausnahme vom Typ „Recoverable“ ausgelöst.

Für FB-, FC- und OB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für FB-, FC- und OB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
HandleErrorsWithinBlock	Bool	false	true

Für FB-, FC- und UDT-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für FB-, FC- und UDT-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
LibraryConformanceStatus	String	-	false

Für GRAPH-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für GRAPH-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
AcknowledgeErrorsRequired	Bool	true	false
CreateMinimizedDB	Bool	false	false
ExtensionBlockName	String	-	-
GraphVersion	String	-	false
InitialValuesAcquisition	String	-	-
LanguageInNetworks	String	-	false
LockOperatingMode	Bool	false	false
PermanentILProcessingIn-MANMode	Bool	false	false
SkipSteps	Bool	false	false

Für GRAPH FB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für GRAPH FB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
WithAlarmHandling	Bool	true	false

Für SCL-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für SCL-Bausteine exportiert. Diese Attribute werden basierend auf dem Typ der PLCs exportiert.

Attribut	Typ	Standardwert	Schreibgeschützt
CheckArrayLimits	Bool	false	false
ExtendedStatus	Bool	false	false
DBAccessibleFromOPCUA	Bool	true	false

Für GRAPH-, SCL- und KOP/FUP-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` für GRAPH-, SCL- und KOP/FUP-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
SetENOAutomatically	Bool	-	false

Für Programmbausteine (außer OB), DB und UDT-Bausteine geltende Attribute

Die folgenden Attribute werden für Programmbausteine (außer OB), DB und UDT-Bausteine unter einer Unit mit den ausgewählten `ExportOptions` exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
Access	UnitAccessType	Unpublished	false

Importverhalten für das Attribut 'Access'

	Import unter Unit	Import nicht unter Unit (ohne SWImportOptions.IgnoreUnitAttributes)	Import nicht unter Unit (mit SWImportOptions.IgnoreUnitAttributes)
XML-Export unter Unit	'Access' verwendet und festgelegt	RecoverableException ausgelöst	'Access' ignoriert
XML-Export nicht unter Unit	'Access' erhält Standardwert	'Access' ist nicht vorhanden	'Access' ist nicht vorhanden

Programmcode

Um einen Baustein ohne Knowhow-Schutz in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a regular block
private static void ExportRegularBlock(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    plcBlock.Export(new FileInfo(string.Format(@"D:\Samples\{ 0 }.xml", plcBlock.Name)),
        ExportOptions.WithDefaults);
}
```

6.4.2.14 Bausteine/UDT mit offenem Verweis importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Der PLC ist nicht online.

Einleitung

Mit TIA Portal Openness können Sie Bausteine und UDTs auch dann importieren, wenn ein zugehöriges Objekt fehlt. Den neuen Modus können Sie dank einer neuen Überladung der entsprechenden Import-Methode verwenden. Die neue Überladung weist einen zusätzlichen Parameter auf, der einen Wert der neuen markierten Enumeration SWImportOption akzeptiert.

Die Openness-Schnittstelle unterstützt den neuen Importmodus unter den folgenden Bedingungen:

Import von	Objektverweis
UDT	UDT
DB (global)	UDT
IDBofUDT	UDT
IDBofFB	FB
ArrayDB	Array von UDT
FB	UDT (Schnittstelle), Multiinstanz
FC	UDT (Schnittstelle)

Programmcode

Um den Import zu ermöglichen, können Sie `SWImportOptions.IgnoreMissingReferencedObject` auch dann verwenden, wenn das Objekt des Verweises fehlt.

```
[Flagged] Enum SWImportOptions
{
None = 0,
IgnoreStructuralChanges = 1,
IgnoreMissingReferencedObjects = 2
}
private static void Main(string[] args)
{
//... // All kinds of blocks
FileInfo file = ...;
PlcBlockComposition.Import(file, ImportOptions.None,
SWImportOptions.IgnoreMissingReferencedObjects);
//...
//... // UDTs
PlcTypeComposition.Import(file, ImportOptions.None,
SWImportOptions.IgnoreMissingReferencedObjects);
//...
}
```

Siehe auch

[Projekt öffnen \(Seite 131\)](#)

6.4.2.15 Bausteine/UDT für Strukturänderungsobjekte importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Der PLC ist nicht online.

Anwendung

Mit der Openness API können Sie Bausteine und UDTs auch dann importieren, wenn aufgrund einer Strukturänderung an zugehörigen Objekten Instanzdaten verloren gegangen sind.

Die Openness-Schnittstelle unterstützt den neuen Importmodus unter den folgenden Bedingungen:

Import von	Objektverweisen
Variable	UDT
UDT	UDT
DB (global)	UDT
IDBofUDT	UDT
IDBofFB	FB
ArrayDB	Array von UDT
FB	UDT (Schnittstelle), Multiinstanz
FC	UDT (Schnittstelle)

Programmcode

Den neuen Modus können Sie dank einer neuen Überladung der entsprechenden Import-Methode verwenden. Die neue Überladung weist einen zusätzlichen Parameter auf, der einen Wert der neuen markierten Enumeration SWImportOptions akzeptiert. Um den Import zu ermöglichen, auch wenn es strukturelle Änderungen gibt und es möglicherweise zu Datenverlust kommt, verwenden Sie "SWImportOptions.IgnoreStructuralChanges".

```
Flagged Enum SWImportOptions
{
  None = 0,
  IgnoreStructuralChanges = 1,
  IgnoreMissingReferencedObjects = 2
}

...
// All kinds of blocks
PlcBlockComposition.Import(file, ImportOptions.None,
  SWImportOptions.IgnoreStructuralChanges);
...
...
// UDTs
PlcTypeComposition.Import(file, ImportOptions.None,
  SWImportOptions.IgnoreStructuralChanges);
...
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.4.2.16 Export/Import des Unit-spezifischen Veröffentlichungsattributs von Bausteinen und Typen

Voraussetzung

- Die Anwendung TIA Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Das Attribut 'Access' ist nur für Bausteine, PLC-Typen und Variablentabellen vorhanden, die sich unter einer Unit befinden. Die XML-Datei des Bausteins, des Typs und der Variablentabelle enthält beim Export in die Unit-Umgebung das Attribut Access

und erhält beim Import den Standardwert Unpublished. Die exportierte XML-Datei des gleichen Bausteins, des gleichen Typs und der gleichen Variablentabelle enthält das Attribut Access in einer Umgebung außerhalb einer Unit nicht

und erhält beim Import keine Werte.

Die allgemeinen XML-Importregeln von Openness lassen den Import von XML-Dateien mit undefinierten Attributen und undefinierten Werten nicht zu. Die Regeln beschränken den Export und Import von XML-Dateien,

die aus einem Objekt in einer Umgebung unterhalb einer Unit stammen, in eine Umgebung außerhalb einer Unit.

Die Regeln beschränken die Nutzung des XML-Exports und -Imports zusätzlich; daher ist die Überladungsmethode import festgelegt, die drei Parameter akzeptiert.

Parameter	Rückgabetyt	Beschreibung
path	String	Gibt den Pfad zur SIMATIC ML-Datei an, die importiert werden soll
importOptions	enum: Siemens.Engineering.ImportOptions	Gibt die beim Import zu verwendende(n) allgemeine(n) Importoption(en) an.
swImportOptions	enum: Siemens.Engineering.SW.SWImportOptions	Gibt die für STEP 7 spezifische(n) Importoption(en) an.

Die Siemens.Engineering.SW.SWImportOptions vom Typ Enum werden durch die folgende neue Importoption erweitert:

- IgnoreUnitAttributes: Gibt an, dass der Importvorgang nicht abgebrochen werden soll, falls in der XML-Datei Unit-bezogene Attribute vorhanden sind und der Import in einer Umgebung außerhalb einer Unit stattfindet.

'IgnoreUnitAttributes' wird berücksichtigt, wenn die XML-Datei

- aus einer Unit exportiert wird
- das Attribut 'Access' enthält
- in eine Umgebung außerhalb einer Unit importiert wird

Wenn die exportierte XML-Datei das Openness-Attribut Access nicht enthält und in eine Unit importiert wird, wird die neue Importoption von der Importlogik nicht berücksichtigt.

Programmcode

```
PlcSoftware plcTarget = GetControllerTargetByPLCName(Session.OpnsProject.Devices, PLCName);
PlcUnitProvider plcUnitProvider = plcTarget.GetService<PlcUnitProvider>();
PlcSoftware plcSoftware = plcTarget.GetService<SoftwareContainer>() as PlcSoftware;
PlcUnit plcUnit1 = plcUnitProvider.UnitGroup.Units[0];
//assuming Unit_1 is already existing
PlcUnit plcUnit2 = plcUnitProvider.UnitGroup.Units[1];
//assuming Unit_2 is already existing
PlcBlock block1 = plcUnit1.BlockGroup.Blocks.Find("Block_1");
//assuming Block_1 is already existing under Unit_1
PlcBlock block2 = plcUnit2.BlockGroup.Blocks.Find("Block_2");
//assuming Block_2 is already existing under Unit_2
PlcBlock block3 = plcSoftware.BlockGroup.Blocks.Find("Block_3");
//assuming Block_3 is already existing under the PLC
```

Programmcode: Objekt wird aus einer Unit exportiert und in eine Unit importiert

Die folgenden Beispiele zeigen das Verhalten der neuen Importoption anhand von Bausteinen. Dasselbe Verhalten gilt jedoch auch für PLC-Typen und Variablentabellen.

Ändern Sie folgenden Programmcode, um einen Baustein zu exportieren, bei dem das Attribut 'Access' auf den Wert 'Unpublished' (Standardwert) gesetzt ist sowie die ExportOptions.WithDefaults und die SWImportOptions.None lauten:

```
block1.SetAttribute("Access", UnitAccessType.Unpublished);
block1.Export(new FileInfo("somepath"), ExportOptions.WithDefaults);
block1.Delete();
plcUnit2.BlockGroup.Blocks.Import(new FileInfo("somepath"), ImportOptions.None,
SWImportOptions.None);
```

Ändern Sie folgenden Programmcode, um einen Baustein zu exportieren, bei dem das Attribut 'Access' auf den Wert 'Unpublished' (Standardwert) gesetzt ist sowie die ExportOptions.WithDefaults und die SWImportOptions.IgnoreUnitAttributes lauten:

```
block1.SetAttribute("Access", UnitAccessType.Unpublished);
block1.Export(new FileInfo("somepath"), ExportOptions.WithDefaults);
block1.Delete();
plcUnit2.BlockGroup.Blocks.Import(new FileInfo("somepath"), ImportOptions.None,
SWImportOptions.IgnoreUnitAttributes);
```

6.4 Daten eines PLC-Geräts importieren/exportieren

Ändern Sie folgenden Programmcode, um einen Baustein zu exportieren, bei dem das Attribut 'Access' auf den Wert 'Published' gesetzt ist sowie die ExportOptions.None und die SWImportOptions.None lauten:

```
block1.SetAttribute("Access", UnitAccessType.Published);  
block1.Export(new FileInfo("somepath"), ExportOptions.None);  
block1.Delete();  
plcUnit2.BlockGroup.Blocks.Import(new FileInfo("somepath"), ImportOptions.None,  
SWImportOptions.None);
```

Ändern Sie folgenden Programmcode, um einen Baustein zu exportieren, bei dem das Attribut 'Access' auf den Wert 'Published' gesetzt ist sowie die ExportOptions.None und die SWImportOptions.IgnoreUnitAttributes lauten:

```
block1.SetAttribute("Access", UnitAccessType.Published);  
block1.Export(new FileInfo("somepath"), ExportOptions.None);  
block1.Delete();  
plcUnit2.BlockGroup.Blocks.Import(new FileInfo("somepath"), ImportOptions.None,  
SWImportOptions.IgnoreUnitAttributes);
```

Hinweis

In allen obigen Programmcode-Abschnitten wird keine Ausnahme ausgelöst und der Importvorgang ist erfolgreich.

Programmcode: Objekt wird aus einer Unit exportiert und in eine Umgebung außerhalb einer Unit importiert

Ändern Sie folgenden Programmcode, um einen Baustein zu exportieren, bei dem das Attribut 'Access' auf den Wert 'Unpublished' (Standardwert) gesetzt ist sowie die ExportOptions.WithDefaults und die SWImportOptions.None lauten:

```
block1.SetAttribute("Access", UnitAccessType.Unpublished);  
block1.Export(new FileInfo("somepath"), ExportOptions.WithDefaults);  
block1.Delete();  
plcUnit2.BlockGroup.Blocks.Import(new FileInfo("somepath"), ImportOptions.None,  
SWImportOptions.None);
```

Hinweis

Im obigen Programmcode ist der Importvorgang nicht erfolgreich und es wird eine wiederherstellbare Ausnahme ausgelöst.

Ändern Sie folgenden Programmcode, um einen Baustein zu exportieren, bei dem das Attribut 'Access' auf den Wert 'Unpublished' (Standardwert) gesetzt ist sowie die ExportOptions.WithDefaults und die SWImportOptions.IgnoreUnitAttributes lauten:

```
block1.SetAttribute("Access", UnitAccessType.Published);  
block1.Export(new FileInfo („somepath"), ExportOptions.None);  
block1.Delete();  
plcUnit2.BlockGroup.Blocks.Import(new FileInfo("somepath"), ImportOptions.None,  
SWImportOptions.None);
```

Hinweis

Im obigen Programmcode ist der Importvorgang erfolgreich und es wird keine Ausnahme ausgelöst.

Ändern Sie folgenden Programmcode, um einen Baustein zu exportieren, bei dem das Attribut 'Access' auf den Wert 'Published' gesetzt ist sowie die ExportOptions.None und die SWImportOptions.None lauten:

```
block1.SetAttribute("Access", UnitAccessType.Published);  
block1.Export(new FileInfo („somepath"), ExportOptions.None);  
block1.Delete();  
plcUnit2.BlockGroup.Blocks.Import(new FileInfo("somepath"), ImportOptions.None,  
SWImportOptions.None);
```

Hinweis

Im obigen Programmcode ist der Importvorgang erfolgreich und es wird eine wiederherstellbare Ausnahme ausgelöst.

Ändern Sie folgenden Programmcode, um einen Baustein zu exportieren, bei dem das Attribut 'Access' auf den Wert 'Published' gesetzt ist sowie die ExportOptions.None und die Importoptionen SWImportOptions.IgnoreUnitAttributes lauten:

```
block1.SetAttribute("Access", UnitAccessType.Published);  
block1.Export(new FileInfo („somepath"), ExportOptions.None);  
block1.Delete();  
plcUnit2.BlockGroup.Blocks.Import(new FileInfo("somepath"), ImportOptions.None,  
SWImportOptions.IgnoreUnitAttributes);
```

Hinweis

Im allen obigen Programmcode-Abschnitten wird keine Ausnahme ausgelöst und der Importvorgang ist erfolgreich.

Objekt wird aus Umgebung außerhalb einer Unit exportiert und in eine Unit importiert

Die exportierte XML-Datei enthält das Openness-Attribut 'Access' nicht; beim Import erhält das Attribut den Standardwert 'Unpublished'.

Objekt wird aus einer Umgebung außerhalb einer Unit exportiert und in eine Umgebung außerhalb einer Unit importiert

Die exportierte XML-Datei enthält das Openness-Attribut 'Access' nicht; beim Import der Datei geschieht nichts.

6.4.2.17 Instanz-DB erstellen**Voraussetzung**

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt wird geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Instanz-DBs für Programmiersprachen wie SCL, KOP, FUP, AWL, Graph und CEM erzeugen.

Programmcode: Instanz-DB für SCL, KOP, FUP, AWL, Graph, & CEM-Baustein erstellen

```
PlcSoftware plc = ...;
// To create instance DB of SCL block
plc.BlockGroup.Blocks.CreateInstanceDB("ConveyerDB", true, 5, "Conveyer_SCL_Block");
// To create instance DB of LAD block
plc.BlockGroup.Blocks.CreateInstanceDB("RelayDB", true, 6, "Relay_LAD_Block");
// To create instance DB of FBD block
plc.BlockGroup.Blocks.CreateInstanceDB("SensorDB", true, 6, "Sensor_FBD_Block");
// To create instance DB of STL block
plc.BlockGroup.Blocks.CreateInstanceDB("ReadIODB", true, 6, "ReadIO_STL_Block");
// To create instance DB of Graph block
plc.BlockGroup.Blocks.CreateInstanceDB("ConveyerDB", true, 6, "Conveyer_Graph_Block");
// To create instance DB of CEM block
plc.BlockGroup.Blocks.CreateInstanceDB("RelayDB", true, 6, "Relay_CEM_Block");
```

6.4.2.18 Auf DB-Wertparameter ohne Export zugreifen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie die folgenden Werte eines Mitgliedsattributs in einem beliebigen DB lesen und schreiben.

Name der Eigenschaft	Datentyp	Modelliert/dynamisch	Zugriff
Name	String	Modelliert	Lesen
StartValue	String	Dynamisch	Lesen/Schreiben
AssignedPro-DiagFB	String	Dynamisch	Lesen
ExternalAccessible	bool	Dynamisch	Lesen/Schreiben
ExternalVisible	bool	Dynamisch	Lesen/Schreiben
ExternalWritable	bool	Dynamisch	Lesen/Schreiben
Retain	bool	Dynamisch	Lesen/Schreiben
SetPoint	bool	Dynamisch	Lesen/Schreiben
DataTypePoint	bool	Dynamisch	Lesen/Schreiben
Snapshot	bool	Dynamisch	Lesen
DefaultValue	bool	Dynamisch	Lesen

Programmcode

Ändern Sie folgenden Programmcode, um auf die Mitgliedsdetails der Datenbaustein-Schnittstelle zuzugreifen:

```
public static void PrintMemberAttributes( plcBlockInterface)
{
    foreach(Member member in plcBlockInterface.Members)
    {
        Console.WriteLine($"Name: {member.Name}");
        Console.WriteLine($"StartValue: {member.GetAttribute("StartValue")}");
        Console.WriteLine($"ExternalAccessible: {member.GetAttribute("ExternalAccessible")}");
        Console.WriteLine($"ExternalVisible: {member.GetAttribute("ExternalVisible")}");
        Console.WriteLine($"ExternalWritable: {member.GetAttribute("ExternalWritable")}");
        Console.WriteLine($"Retain: {member.GetAttribute("Retain")}");
        Console.WriteLine($"Setpoint: {member.GetAttribute("Setpoint")}");
        Console.WriteLine($"DataTypeName: {member.GetAttribute("DataTypeName")}");
        Console.WriteLine($"Snapshot: {member.GetAttribute("Snapshot")}");
        Console.WriteLine($"DefaultValue: {member.GetAttribute("DefaultValue")}");
    }
}
```

Ändern Sie folgenden Programmcode, um den Anfangswert aus dem DB zu lesen:

```
PlcBlockInterface bi = dbbblock.Interface;
MemberComposition members = bi.Members;
Member member = members.Find("Room_Temperature");
string startValue = member.StartValue;
//Normal get attribute should be possible as usual
startValue = member.GetAttribute("StartValue");
//Array
initialSpeedvar = members.Find("Motor.InitialSpeed[0]");
object motorInitialSpeed = initialSpeedvar.StartValue;
//UDT-struct
axisSpeedvar = members.Find("FillingStation.Conveyer.AxisSpeed");
object axisSpeed = axisSpeedvar.StartValue;
axisSpeed = axisSpeedvar.GetAttribute("StartValue");
//Struct
initialSpeedvar = paramF.Find("DischargeValve.FlowMeter.InitialSpeed[0]");
object initialSpeed = initialSpeedvar.StartValue;
initialSpeed = initialSpeedvar.GetAttribute("StartValue");
```


Ändern Sie folgenden Programmcode, um den Anfangswert in den DB zu schreiben:

```
PlcBlockInterface bi = dbbblock.Interface;
MemberComposition members = bi.Members
Member member = members.Find("Room_Temperature");
member.StartValue = 10.2;
//Normal set attribute should be possible as usual
member.SetAttribute("StartValue", 20.3);
//Array
initialSpeedvar = members.Find("Motor.InitialSpeed[0]");
initialSpeedvar.StartValue = 36;
initialSpeedvar.SetAttribute("StartValue", 56);
//UDT-struct
axisSpeedvar = members.Find("FillingStation.Conveyer.AxisSpeed");
object axisSpeed = axisSpeedvar.StartValue;
axisSpeed.SetAttribute("StartValue", 40.5);
initialSpeedvar = paramF.Find("DischargeValve.FlowMeter.InitialSpeed[0]");
object initialSpeed = initialSpeedvar.StartValue;
initialSpeed.SetAttribute("StartValue", 12);
```

Ändern Sie folgenden Programmcode, um den Anfangswert einer Variablen auf 'myTest' einzustellen:

```
PlcSoftware myPlcSoftware =
tiaProject.Devices[0].DeviceItems[1].GetService<SoftwareContainer>().Software as
PlcSoftware;
DataBlock myDatablock = myPlcSoftware.BlockGroup.Blocks[1] as DataBlock;
myDatablock.Interface.Members[0].SetAttribute("StartValue", "'myTest'");
```

Siehe auch

Projekt öffnen (Seite 131)

6.4.2.19 Export/Import von PLC-Meldetextlisten

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Sie können mit TIA Portal Openness PLC-Meldetextlisten exportieren und importieren. Der Export/Import erfolgt im Format XLSX.

Struktur der Excel-Datei

Die generierte XLSX-Datei besteht aus zwei Tabellenblättern – einem für die Textlisten und einem für die Einträge. Die erstellte XLSX-Datei empfängt bei dem Vorgang eine benutzerdefinierte Eigenschaft namens FileContent mit dem Wert "Alarm text lists". Wenn diese Eigenschaft fehlt oder ungültig ist (keinen oder einen abweichenden Wert enthält), wird der Import abgelehnt.

Die Daten der beiden Arbeitsblätter sind miteinander verknüpft. Dabei verweist das übergeordnete TextListEntry.Parent auf TextList.Name

	A	B	C	D	E
1	Name	ListRange	Comment [en-US]	Comment [de-DE]	Comment [fr-FR]
2	NextTextList	Decimal	English	Deutsch	French

A	B	C	D	E	F
Parent	From	To	Text [de-DE]	Text [en-US]	Text [fr-FR]
NextTextList	0	1	xyzxyzxyz	rstrstrst	uvwuvwuvw

Modell des Engineering-Objekts

Die Definition des Engineering-Objektmodells für den neu eingeführten Typ PlcAlarmTextListProvider. Der Zugriff darauf erfolgt über die Engineering-Objekte PlcUnit und PlcSoftware.

Exportieren

Für den Exportvorgang stehen die zwei API-Methoden zur Verfügung. Wenn alle Textlisten einer PLC-/Software Unit in allen aktivierten Projektsprachen exportiert werden sollen, ist der einzige benötigte Parameter "path".

Er definiert den Speicherort, an dem sich die XLSX-Ergebnisdatei nach erfolgreichem Export befinden soll.

```
FileInfo fileInfo = new FileInfo(Path.Combine(Environment.CurrentDirectory,
    $"{xlsxName}.xlsx"));
PlcAlarmTextListProvider textListProvider = GetTextListProvider(unitName);
textListProvider.ExportToXlsx(fileInfo);
```

Wenn nicht alle Textlisten einer PLC-/Software Unit exportiert werden müssen oder wenn die Sprachen – für die der Textlistenexport vorgesehen ist – gefiltert werden sollen, ist in der API eine andere Methode verfügbar.

Damit werden die zu exportierenden Textlisten über die Liste der Textlistenamen angegeben und die Sprachen, für die der Export erfolgen soll, über die Liste der Sprachen angegeben.

Beispiel für die Verwendung der Aktion ExportToXlsx:

6.4 Daten eines PLC-Geräts importieren/exportieren

```

private void ExportPlcOrUnit(
    string stationName,
    string plcName,
    string xlsxName,
    string[] textListNames,
    string[] cultureCodes,
    string unitName = null)
{
    LanguageSettings languageSettings = Project.LanguageSettings;
    LanguageComposition supportedLanguages = languageSettings.Languages;
    IEnumerable<Language> cultureInfos = null;
    if (cultureCodes != null)
    {
        cultureInfos = cultureCodes.Select(i =>
            supportedLanguages.Find(CultureInfo.GetCultureInfo(i)));
    }
    FileInfo fileInfo = new FileInfo(Path.Combine(Environment.CurrentDirectory,
        $"{xlsxName}.xlsx"));
    PlcAlarmTextListProvider textListProvider = GetTextListProvider(stationName, plcName,
        unitName);
    TextListXlsxResult result = null;
    string exceptionMessage = string.Empty;
    ExceptionMessageData? userExceptionMessageData = null;
    try
    {
        result = textListProvider.ExportToXlsx(fileInfo, textListNames, cultureInfos);
    }
    catch (EngineeringTargetInvocationException e)
    {
        exceptionMessage = e.ToString();
        userExceptionMessageData = e.DetailMessageData.FirstOrDefault();
    }
}

private PlcAlarmTextListProvider GetTextListProvider(string stationName, string plcName,
    string unitName = null)
{
    DeviceComposition devices = Project.Devices;
    Device device = devices.Where(station => station.Name == stationName).FirstOrDefault();
    if (device == null)
    {
        throw new InvalidOperationException($"The requested '{stationName}' station is not found");
    }
    PlcSoftware plcTarget = (PlcSoftware)FindSoftwareTarget(device, plcName);
    if (plcTarget == null)
    {
        throw new InvalidOperationException($"The requested '{plcName}' PLC SW is not found")
    }
    if (unitName != null)
    {
        PlcUnitProvider unitProvider = plcTarget.GetService<PlcUnitProvider>();
        PlcUnit unit = unitProvider.UnitGroup.Units.Where(unit => unit.Name ==
            unitName).FirstOrDefault();
        if (unit == null)
        {
            throw new InvalidOperationException($"The requested '{unitName}' unit is not found");
        }
    }
}

```

```

}
return unit.GetService<PlcAlarmTextListProvider>();
}
return plcTarget.GetService<PlcAlarmTextListProvider>();
}

```

Fehlerbehandlung:

- Wenn die angegebene Sprache nicht als Projektsprache aktiviert ist, wird die `UserException` ausgelöst. Beispielsweise kann "Erforderliche Sprache 'Konkani (Indien)' nicht exportiert werden, weil es sich dabei nicht um eine im Projekt verwendete Sprache handelt. Die gültigen Sprachen sind dagegen 'Deutsch (Deutschland)', 'Englisch (USA)'"
- Wenn keine Anwendertextliste vorhanden ist, wird eine `UserException` (`TextListNotFoundException`) mit folgender Meldung ausgelöst: "Zu folgendem Element ist keine Textliste vorhanden: <PLCName>."
- Wenn ein bestimmter Textlistenname nicht vorhanden ist oder wenn er vorhanden ist, aber keine Anwendertextliste dazu existiert, wird eine `UserException` ausgelöst. Beispiele:
"Textliste Benutzer_1 wird unter PLC_1 nicht gefunden."
"Textliste SYSTEM_SDiag_CmpCpuName kann nicht exportiert werden, da es sich nicht um eine Anwendertextliste handelt."

Importieren

Der folgende Importvorgang ist unter der Klasse `PLCArmTextListProvider` verfügbar. Mit Hilfe des Parameters `importOptions` kann angegeben werden, ob beim Import vorhandene Textlisten überschrieben werden sollen. Wenn ...

der Parameter `importOptions` auf `ImportOptions.None` gesetzt ist und beim Import eine bereits vorhandene Textliste aktualisiert werden soll, wird eine `UserException` ausgelöst. Wurde der Parameter `importOptions` auf

`ImportOptions.Override` gesetzt, werden die bereits vorhandenen Textlisten beim Import aktualisiert.

Vorhandene Textlisten werden über ihren Namen identifiziert. Neue Daten (Textliste, Eintrag) werden an vorhandene Daten angehängt. Vorhandene Textlisten werden nur gelöscht, wenn es in der Importdatei entsprechende Daten gibt.

Enthält also die Excel-Datei Textlisteneinträge für eine bereits vorhandene Textliste, werden die vorhandenen Einträge gelöscht und die in der Excel-Datei enthaltenen Einträge werden in das TIA-Projekt für die

betreffende Textliste importiert (dabei werden keine Textlisteneinträge zusammengeführt).

Wenn die Datei ungültige Einträge enthält, wird die Textliste mit den ungültigen Einträgen nicht importiert. (Beispiele für ungültige Einträge: Überschneidung zwischen Einträgen oder Datentyp abweichend vom Datentyp der Textliste, die die Einträge enthält).

Die Texte werden nur in den Sprachen importiert, die im Projekt aktiv sind. Wenn die Datei mehr Sprachen als das Projekt enthält, wird der Text in den zusätzlichen Sprachen nicht importiert und eine Warnung

wird protokolliert. Wenn das Projekt mehr Sprachen als die Datei umfasst, werden Texte, für die keine entsprechenden Daten in der Datei vorhanden sind, nicht bearbeitet (oder bleiben leer, falls beim Import neue Textlisten oder Textbereiche eingefügt werden).

In ein schreibgeschütztes Projekt können keine Daten importiert werden. Falls ein Benutzer versucht, Daten in ein schreibgeschütztes Projekt zu importieren, müsste eine `UserException` ausgelöst werden.

Ein Import ist ein komplexer Vorgang, bei dem verschiedenartige Fehler auftreten können. Wenn der Fehler kein kritischer Fehler ist, wird der Importvorgang beendet und die Methode gibt ein Objekt `TextListXlsxResult` zurück, das

den Pfad zur Protokolldatei enthält (die wiederum die während des Imports erzeugten Meldungen enthält) und das den Status `TextListXlsxResultState.Warning` hat.

Wenn der Fehler ein kritischer Fehler ist, wird eine `UserException` ausgelöst, die Details zum Fehler enthält. Handelt es sich um einen fatalen Fehler, wird eine `NonRecoverableException` ausgelöst.

```
<?xml version="1.0" standalone="no"?>
<?xml-stylesheet type='text/xsl' href='MassDataHandlerLogFile.xsl'?>
<LogFile
  titleName="PLCAlarmTextLists_InvalidLanguage.xlsx__2019.06.07_13.46.45.070__Import_Log.xml"
  projectName="D:\TIA\dev\WM5_WinCC_HW_Work\binaries\Debug\x64\Tests\Siemens.Simatic.AlarmSe
  rvices.Integration.Test.Openness\TextListXlsxFiles\PLCAlarmTextLists_InvalidLanguage.xlsx"
  typeText="Type" messageText="Message" timeText="Time">
  <LogEntry type="Warning" dateTime="3:46:45 PM">
    <Message>The language ID in column 'Comment [abcd-EF]' is missing or is invalid (sheet
    'TextList'). The texts in this language are not imported.</Message>
  </LogEntry>
  <LogEntry type="Warning" dateTime="3:46:45 PM">
    <Message>The language ID in column 'Text [abcd-EF]' is missing or is invalid (sheet
    'TextListEntry'). The texts in this language are not imported.
  </Message>
  </LogEntry>
  <LogEntry type="Information" dateTime="3:46:45 PM">
    <Message>Import completed: 2 text lists with 9 entries.</Message>
  </LogEntry>
</LogFile>
```

Hinweis

- Für die XLSX-Dateien, die beim Export durch TIA Portal V16 erzeugt werden, müsste die benutzerdefinierte Eigenschaft `FileVersion` mit dem Wert 1 angegeben werden.
 - Für die XLSX-Dateien, die beim Export durch TIA Portal V16 erzeugt werden, müsste die benutzerdefinierte Eigenschaft `FileContent` mit dem Wert `Alarm text lists` angegeben werden.
 - Die Protokolldatei wird während des Importvorgangs im Ordner "Protokolle" des TIA-Projekts erzeugt.
-

Ergebnis

Das Objekt `TextListXlsxResult` enthält Informationen über das Ergebnis des Exports bzw. Imports. Es umfasst ein `FileInfo`-Attribut namens `LogFilePath`, das auf die Protokolldatei für den beendeten Vorgang verweist. Als Status

(vom Typ `TextListXlsxResultState`) wird der Endzustand des Vorgangs angezeigt, der entweder OK, "Warning" oder "Error" lautet. Beim Ergebnis "Error" ist der Vorgang fehlgeschlagen, bei "Warning" war der Vorgang teilweise erfolgreich.

6.4.2.20 Dokumentinformation in eine SimaticML-Datei exportieren

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit dem TIA Portal können Sie die Typen `PlcTag`, `PlcTagTable` und `PlcUserConstant` in eine SimaticML-Datei mit benutzerspezifischen Dokumentinformationen exportieren. Die aus dem TIA Portal in die SimaticML-Datei exportierten und aus der SimaticML-Datei in das TIA Portal importierten Dokumentinformationen sind nicht relevant.

Beim neuen Export können Sie entweder die vollständigen Dokumentinformationen beim Export der Openness-Datentypen wie `PlcTag`, `PlcTagTable` und `PlcUserConstant` ignorieren oder selbst einstellen, welche Dokumentinformationen in der SimaticML-Datei zur Verfügung stehen müssen.

DocumentInfoOptions

Mit Hilfe von `Siemens.Engineering.DocumentInfoOptions` können Sie festlegen, welche Dokumentinformationen in die SimaticML-Datei exportiert werden sollen. `DocumentInfoOptions` ist ein Enum-Merker, über den Sie mehrere Konfigurationen auswählen können. Die folgenden Konfigurationen stehen für den Export zur Verfügung:

Enum	Beschreibung
None	Keine Dokumentinfo wird exportiert
ExportSetting	Daten nur mit Dokumentinformation <code>ExportSetting</code> exportieren; enthält beispielsweise die Details der für den Export verwendeten <code>ExportOption</code> .
InstalledProducts	Daten nur mit Dokumentinformation <code>InstalledProducts</code> exportieren; enthält beispielsweise Produktnamen und Namen von Optionspaketen.
CreatedTimeStamp	Daten nur mit Dokumentinformation <code>CreatedTimeStamp</code> exportieren
All	Daten mit allen Dokumentinformationen exportieren (<code>CreatedTimeStamp</code> , <code>ExportSetting</code> , <code>InstalledProducts</code>)

Programmcode: Export mit DocumentInfoOptions

```
PlcTagTable tagTable = FindTagTableToBeExported("TestTagTable");
FileInfo testTagTableExportFile = new FileInfo($"E:\\temp\\TestTagTable.xml");
// No document info will be exported since the configuration is None.
tagTable.Export(testTagTableExportFile, ExportOptions.WithDefaults,
DocumentInfoOptions.None);
FileInfo testTagExportFile = new FileInfo($"E:\\temp\\TestTag.xml");
PlcTag tag = tagTable.Tags.Find("TestTag");
// CreatedTimeStamp and InstalledProduct will be exported as DocumentInfo.
tag.Export(testTagExportFile, ExportOptions.WithDefaults,
DocumentInfoOptions.CreatedTimeStamp | DocumentInfoOptions.InstalledProducts);
FileInfo testUserConstantExportFile = new FileInfo($"E:\\temp\\TestUserConstant.xml");
PlcUserConstant plcUserConstant = tagTable.UserConstants.Find("TestUserConstant");
// Full Document info will be exported
plcUserConstant.Export(testUserConstantExportFile, ExportOptions.WithDefaults,
DocumentInfoOptions.All);
```

6.4.2.21 Export/Import von Meldungsinstantztexten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie Meldungsinstantztexte exportieren und importieren. Die Option ähnelt dem Export und Import von Typmeldetexten über den Editor "PLC-Überwachungen und Meldungen" und ebenfalls dem Export/Import von Instanzmeldetexten über den CFC-Meldeeditor. Die Funktion des Exports und Imports von Meldungsinstantztexten über TIA Portal Openness soll für Übersetzungszwecke verwendet werden. Die Funktion erstellt beim Import keinen Meldungsinstantztyp und keine Meldeklasse. Beispiel: Ist die über den Namen der Excel-Datei identifizierte Meldeklasse im Zielprojekt vorhanden, wird sie beim Import für die Instanzmeldung festgelegt; eine nicht vorhandene Klasse wird jedoch nicht erstellt.

Struktur der Excel-Datei

	A	B	C	D	E
1	Location	Alarm name	Alarm class	"Alarm text" - English (United States) / [en-US] / Event text	Fieldinfo / "Alarm text" / Event text
2	Datenbaustein_1	Static_1	### Inherited from Type ###	alarm text 1	"Info text" - Eng
3					English Info Inst
4					
5					

Bei "Inherited from Type ###" handelt es sich um einen Spezialinhalt, der das System anweist, den Text (bzw. die Meldeklasse) des Typs zu verwenden. Das Feld "Alarm class"

bezieht sich auf einen im Zielprojekt vorhandenen Namen einer Meldeklasse. Eine wichtige Regel lautet, dass bei einem bestimmten Meldetext entweder alle Sprachfelder ###Inherited from Type### enthalten oder keines. Bei Feldern mit gemischtem Inhalt werden keinerlei Texte für diese Meldung importiert und es wird eine Warnung ausgegeben.

Die Funktion Exportieren sorgt für die Dateidatenkonsistenz.

Frei überschreibbare Felder sind sprachabhängig; die Kopfzeilen dieser Felder enthalten Sprachkennungen wie beispielsweise "Alarm text" – English (United States) / [en-US] / Event text.

Darüber hinaus enthält die Datei Spezialeigenschaften, über die das System vor dem Import Überprüfungen durchführen kann. Sind diese Eigenschaften ungültig, kann der Import nicht durchgeführt werden.

Export

Die folgende Aktion ist unter PlcAlarmTextProvider für den Export verfügbar.

- PlcAlarmTextProvider

Die folgenden Parameter werden unter PlcAlarmTextProvider für den Import bereitgestellt:

- Pfad
- Sprache
- Option

Der Export kann eine PLC oder eine Software Unit sein. Der Pfadparameter ist obligatorisch, Sprachen können null oder leer sein. In diesem Fall werden Texte in allen aktiven Projektsprachen exportiert. Der Optionsparameter ist ebenfalls obligatorisch. Er nimmt einen oder mehrere Werte der Enumeration PlcAlarmTextXlsxExportOption an.

PlcAlarmTextXlsxExportOption hat die folgenden Werte:

- None (nur Meldetexte werden exportiert)
- IncludeInfoText (auch Infotexte werden exportiert)
- IncludeAdditionalTexts (Export enthält außerdem Zusatztext 1-9)
- IncludeAlarmClass (Einstellung der Meldeklasse wird dem Export hinzugefügt)
- All

Die drei obigen Include-Optionen können miteinander kombiniert werden. Sollen beispielsweise der Meldetext, Zusatztexte und die Meldeklasse, nicht jedoch der Infotext exportiert werden, setzen Sie IncludeAdditionalTexts und IncludeAlarmClass.

Im Falle eines Fehlers wird eine UserException ausgelöst. Dabei kann es sich beispielsweise um inaktive bzw. nicht vorhandene Sprachen oder Datei-E/A-Fehler handeln. Die Meldung zur Anzeige einer ungültigen Sprache weist das folgende Format auf:

"Erforderliche Sprache 'Konkani (Indien)' kann nicht exportiert werden, weil es sich dabei nicht um eine in diesem Projekt verwendete Sprache handelt. Gültige Sprachen sind dagegen 'Deutsch (Deutschland)', 'Englisch (USA)'"

Import

Die folgende Aktion ist unter PlcAlarmTextProvider für den Import verfügbar:

- ImportInstanceTextsToXlsx

Die folgenden Parameter werden unter PlcAlarmTextProvider für den Import bereitgestellt:

- Pfad
- Sprache

Der Import kann eine PLC oder eine Software Unit sein. Der Pfadparameter ist obligatorisch, Sprachen können null oder leer sein. In diesem Fall werden Texte in allen aktiven Projektsprachen importiert. (Enthält die Excel-Datei im Projekt inaktive Sprachen, werden diese Sprachen nicht aktiviert und es erfolgt kein Import in den jeweiligen Sprachen.)

```
FileInfo fileInfo = new FileInfo(Path.Combine(s_XlsxFilesFolderName, $"{xlsxName}.xlsx"));
List<Language> cultureInfos = new List<Language>();
PlcSoftware plcTarget = ...;
PlcAlarmTextProvider alarmTextsProvider = plcTarget.GetService<PlcAlarmTextProvider>();
PlcAlarmTextXlsxResult result = alarmTextsProvider.ExportInstanceTextsToXlsx(fileInfo,
cultureInfos, PlcAlarmTextXlsxExportOption.All);
//Project project = ...;
LanguageSettings languageSettings = project.LanguageSettings;
LanguageComposition supportedLanguages = languageSettings.Languages;
cultureInfo = cultureInfos.Add(supportedLanguages.Find(CultureInfo.GetCultureInfo("en-US")));
PlcAlarmTextProvider alarmTextsProvider = plcTarget.GetService<PlcAlarmTextProvider>();
result = alarmTextsProvider.ImportInstanceTextsFromXlsx(fileInfo, cultureInfo);
```

Ergebnis

Das Objekt PlcAlarmTextXlsxResult bietet Informationen über das Ergebnis des Exports bzw. Imports. Es umfasst eine FileInfo-Struktur namens LogFilePath, die auf die Protokolldatei für den beendeten Vorgang verweist. Als Status (vom Typ PlcAlarmTextXlsxResultState) wird der Endzustand des Vorgangs angezeigt. Dessen Wert lautet "OK", "Warning" oder "Error". Beim Ergebnis "Error" ist der Vorgang fehlgeschlagen, bei "Warning" war der Vorgang teilweise erfolgreich.

Ausnahme

Nicht wiederherstellbare Ausnahmen

Dabei handelt es sich um technische Probleme, die Ihnen nicht angezeigt werden sollten.

- "Missing PlcAlarmTextProvider"
- "Plc/SW Unit cannot be retrieved"
- Parameter wurde nicht gefunden. Wenden Sie sich an den technischen Support von Siemens.
- "ProjectService is not available"

- Pfadparameter wurde nicht gefunden. (Parameter: nnn, mmm) Wenden Sie sich an den technischen Support von Siemens.
- Der Ergebniszustand ist zwar nicht "Error", es werden jedoch Fehlermeldungen angezeigt: xyz.

Wiederherstellbare Ausnahmen

Diese könnten z. B. aufgrund von ungültigen Parameterwerten angezeigt werden.

- Die erforderliche Sprache 'xx-XX' kann nicht exportiert werden, weil sie nicht für diese PLC festgelegt wurde. Gültige Sprachen sind yy-YY, zz-ZZ.

Es können auch verschiedene Systemfehlermeldungen angezeigt werden, z. B. zu Datei-E/A-Fehlern usw.

Siehe auch

Projekt öffnen (Seite 131)

6.4.2.22 Export/Import von Meldeklassen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Sie können mit TIA Portal Openness Meldeklassen exportieren/importieren. Der Export/Import erfolgt im Format .DAT.

Modell des Engineering-Objekts

Die Definition des Engineering-Objekt-Modells für den neu eingeführten Typ AlarmClassDataProvider.

Der Zugriff darauf ist über das Engineering-Objekt ProjectBase möglich.

Exportieren

Die folgende Aktion ist unter der Klasse AlarmClassDataProvider für den Export verfügbar.

Zur Exportaktion gehört ein Parameter: path.

Mit dem Pfadparameter kann der Dateipfad angegeben werden, in dem das Ergebnis des Exports abgelegt wird. Der Typ des Parameters lautet FileInfo.

Das Ergebnis der Aktion ist ein `AlarmClassExportImportResult` object. Es enthält die Anzahl der Fehler, Warnungen und den Status der Exportergebnisse ("Success", "Warning" oder "Error").

Zur Aktion gehört auch eine `Messages`-Eigenschaft mit Angaben zur Zusammensetzung. Diese umfasst die während des Exports generierten Meldungen zusammen mit der Statusangabe ("Success", "Error" oder "Warning").

Beispiel für die Verwendung der Aktion:

```
FileInfo fileInfo = new FileInfo(@"D:\AlarmClasses.DAT");
AlarmClassDataProvider provider = Project.GetService<AlarmClassDataProvider>();
if (fileInfo.Exists)
{
    fileInfo.Delete();
}
AlarmClassExportImportResult result;
result = provider.Export(fileInfo);
Assert.AreEqual(AlarmClassExportImportResultState.Success, result.State);
Assert.AreEqual(0, result.ErrorCount);
Assert.AreEqual(0, result.WarningCount);
AlarmClassExportImportResultMessageComposition messages = result.Messages;
Assert.AreEqual(1, messages.Count);
Assert.AreEqual($"Export of Alarm class settings file '{fileInfo.FullName}' was successful.", messages.FirstOrDefault().Message);
Assert.AreEqual(AlarmClassExportImportResultState.Success, messages.FirstOrDefault().State);
Assert.IsTrue(File.Exists(fileInfo.FullName), "Exported file does not exist.");
fileInfo.Delete();
```

Fehlerbehandlung:

- Wenn der Gesamtstatus des Exportvorgangs "Error" lautet, wird eine `UserException` ausgelöst und die Liste der Fehlermeldungen wird in der Ausnahme selbst gespeichert.

Clientseitig kann eine Ausnahme `EngineeringTargetInvocationException` abgefangen werden, bei der die Eigenschaft `DetailMessageData` die Fehlermeldungen enthält.

- Mögliche Fehler:
 - E/A-Fehler beim Exportvorgang

Importieren

Der Importvorgang ähnelt der Exportaktion.

Beispiel für die Verwendung der Importaktion:

```
FileInfo fileInfo = new FileInfo(@"D:\AlarmClasses.DAT");
AlarmClassDataProvider provider = Project.GetService<AlarmClassDataProvider>();
AlarmClassExportImportResult result;
result = provider.Import(fileInfo);
Assert.AreEqual(AlarmClassExportImportResultState.Success, result.State);
Assert.AreEqual(0, result.ErrorCount);
Assert.AreEqual(0, result.WarningCount);
AlarmClassExportImportResultMessageComposition messages = result.Messages;
Assert.AreEqual(1, messages.Count);
Assert.AreEqual($"Import of Alarm class settings file '{fileInfo.FullName}' was successful.", messages.FirstOrDefault().Message);
Assert.AreEqual(AlarmClassExportImportResultState.Success, messages.FirstOrDefault().State);
```

Fehlerbehandlung:

- Wenn der Gesamtstatus des Exportvorgangs Error lautet, wird eine `UserException` ausgelöst und die Liste der Fehlermeldungen wird in der Ausnahme selbst gespeichert.

Clientseitig kann eine Ausnahme `EngineeringTargetInvocationException` abgefangen werden, bei der die Eigenschaft `DetailMessageData` die Fehlermeldungen enthält

- Mögliche Fehler
 - Dateierweiterung der zu importierenden Datei ist ungültig
 - Dateierweiterung der zu importierenden Datei fehlt
 - Beim Import würde die Anzahl der Meldeklassen die maximal zulässige Anzahl von Meldeklassen im Projekt übersteigen
 - Die zu importierende Datei verwendet nicht die korrekte Schemaversion (nur Versionen 1.0 und 2.0 werden unterstützt)
 - Es gibt keine Schnittmenge zwischen den aktivierten Projektsprachen und den in der Importdatei angegebenen Sprachen
 - Keine Sprache definiert
 - E/A-Fehler beim Exportvorgang

Hinweis

Der Import von Dateien, die beim Export mit TIA Portal V16 erzeugt wurden, muss unterstützt werden.

Ergebnis

Das Objekt `AlarmClassExportImportResult` enthält die Informationen über das Ergebnis des Exports bzw. Imports. Es umfasst einen Navigator `AlarmClassExportImportResultMessageComposition` namens "Messages", dessen Name auf die beim Export- und Importvorgang generierten Meldungen hinweist. Als Status (vom Typ `AlarmClassExportImportResultState`) wird der Endzustand des Vorgangs angezeigt, der entweder "OK", "Warnung" oder "Fehler" lautet. Beim Ergebnis "Error" ist der Vorgang fehlgeschlagen, bei "Warning" war der Vorgang teilweise erfolgreich.

Siehe auch

Projekt öffnen (Seite 131)

6.4.2.23 Export/Import von globalen Überwachungen des ProDiag-FB

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit TIA Portal verbunden
Siehe Projekt öffnen (Seite 131)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 84)

Einleitung

Mit TIA Portal Openness können Sie den Export/Import von globalen Überwachungen des Funktionsbausteins ProDiag unterstützen. Diese Funktion wird von einem Mechanismus des Dienst-Providers unterstützt. Auf denselben Dienst-Provider greift der Baustein zusammen mit der PLC und der Software Unit bei Ausführung der Import- und Export-Operationen zurück.

Bei Ausführung eines Exports/Imports werden Ihnen als Benutzer der entsprechende Ergebnisstatus oder die entsprechenden Benutzerausnahmen mitgeteilt.

Mit der folgenden Methode können globale Überwachungen des Bausteins ProDiag exportiert und importiert werden:

Methode	Beschreibung
<code>ExportSupervisionsToXlsx(FileInfo path)</code>	Die Überwachungsvariablen einschließlich der Einstellungen in dem vom TIA-Portal unterstützten Format werden exportiert und die Ergebnisdaten des Exports werden erzeugt
<code>ImportSupervisionsFromXlsx(FileInfo path, ImportOptions importOptions)</code>	Überwachungsvariablen werden mit unterstützten Importoptionen importiert
<code>ImportSupervisionSettingsFromXlsx(FileInfo path, ImportOptions importOptions)</code>	Überwachungseinstellungen werden importiert, die in der exportierten Datei zusammen mit überwachten Variablen abgelegt werden

Unter SupervisionXlsxResult sind die folgenden Eigenschaften verfügbar:

Name der Eigenschaft	Datentyp	Zugriff
LogFilePath	System.IO.FileInfo	Lesen
State	SupervisionXlsxResult	Lesen

Der SupervisionXlsxResultState hat folgende Werte:

ENUM	Werte
SupervisionXlsxResultState	Success
	Failure

Programmcode

Ändern Sie den folgenden Programmcode, um globale Überwachungen des ProDiag-Bausteins zu exportieren:

```
// File Path for the export
FileInfo fileInfo = new FileInfo(@"C:\Users\z003jwfc\Desktop\Supervisions_Openness.xlsx");
//SW is nothing but PlcSoftware / PlcUnit.
var proDiagBlock = (FB)SW.BlockGroup.Blocks.Find("Block1");
SupervisionProvider supervisionProvider = proDiagBlock.GetService<SupervisionProvider>();
SupervisionXlsxResult result = supervisionProvider.ExportSupervisionsToXlsx(fileInfo);
```

Ändern Sie den folgenden Programmcode, um globale Überwachungen des ProDiag-Bausteins zu importieren:

```
// File Path for the import
FileInfo fileInfo = new FileInfo(@"C:\Users\z003jwfc\Desktop\SupervisionsOpenness.xlsx");
//SW is nothing but PlcSoftware / PlcUnit.
var proDiagBlock = (FB)SW.BlockGroup.Blocks.Find("Block1");
SupervisionProvider supervisionProvider = proDiagBlock.GetService<SupervisionProvider>();
//import supervisions
SupervisionXlsxResult result = supervisionProvider.ImportSupervisionsFromXlsx(fileInfo,
ImportOptions.None);
SupervisionXlsxResult result = supervisionProvider.ImportSupervisionsFromXlsx(fileInfo,
ImportOptions.Override);
//import supervision settings
SupervisionXlsxResult result =
supervisionProvider.ImportSupervisionSettingsFromXlsx(fileInfo, ImportOptions.None);
SupervisionXlsxResult result =
supervisionProvider.ImportSupervisionSettingsFromXlsx(fileInfo, ImportOptions.Override);
```

6.4.2.24 Export/Import von ProDiag-Überwachungen aus globalem Übersichtseditor

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie den Export und Import von globalen Überwachungen unterstützen, die für eine PLC oder Software Unit verfügbar sind. Diese Funktion wird von einem Mechanismus des Dienst-Providers unterstützt. Auf denselben Dienst-Provider greifen die PLC und die Software Unit bei der Ausführung von Export- und Import-Operationen zurück. Bei Ausführung eines Exports/Imports werden Ihnen als Benutzer das entsprechende Ergebnis oder Ausnahmen mitgeteilt.

Mit der folgenden Methode können globale Überwachungen exportiert und importiert werden:

Methode	Beschreibung
ExportSupervisionsToXlsx(FileInfo path)	Die Überwachungsvariablen einschließlich der Einstellungen in dem vom TIA-Portal unterstützten Format werden exportiert und die Ergebnisdaten des Exports werden erzeugt
ImportSupervisionsFromXlsx(FileInfo path, ImportOptions importOptions)	Überwachungsvariablen werden mit unterstützten Importoptionen importiert
ImportSupervisionSettingsFromXlsx(FileInfo path, ImportOptions importOptions)	Überwachungseinstellungen werden importiert, die in der exportierten Datei zusammen mit überwachten Variablen abgelegt werden

Unter SupervisionXlsxResult sind die folgenden Eigenschaften verfügbar:

Name der Eigenschaft	Datentyp	Zugriff
LogFilePath	System.IO.FileInfo	Lesen
State	SupervisionXlsxResult	Lesen

Der SupervisionXlsxResultState hat die folgenden ENUM-Werte:

ENUM	Werte
SupervisionXlsxResultState	Success
	Failure

Einleitung

Ändern Sie den folgenden Programmcode, um globale Überwachungen zu exportieren:

```
//File Path for the export
FileInfo fileInfo = new FileInfo(@"C:\Users\z003jwfc\Desktop\Supervisions_Openness.xlsx");
//SW is nothing but PlcSoftware / PlcUnit.
SupervisionProvider supervisionProvider = SW.GetService<SupervisionProvider>();
SupervisionXlsxResult result = supervisionProvider.ExportSupervisionsToXlsx(fileInfo);
```

Ändern Sie den folgenden Programmcode, um globale Überwachungen zu importieren:

```
//File Path for the import.
FileInfo fileInfo = new FileInfo(@"C:\Users\z003jwfc\Desktop\Supervisions_Openness.xlsx");
//SW is nothing but PlcSoftware / PlcUnit.
SupervisionProvider supervisionProvider = SW.GetService<SupervisionProvider>();
//import supervisions
SupervisionXlsxResult result = supervisionProvider.ImportSupervisionsFromXlsx(fileInfo,
ImportOptions.None);
SupervisionXlsxResult result = supervisionProvider.ImportSupervisionsFromXlsx(fileInfo,
ImportOptions.Override);
//import supervision settings
SupervisionXlsxResult result =
supervisionProvider.ImportSupervisionsSettingsFromXlsx(fileInfo, ImportOptions.None);
SupervisionXlsxResult result =
supervisionProvider.ImportSupervisionsSettingsFromXlsx(fileInfo, ImportOptions.Override);
```

Siehe auch

Projekt öffnen (Seite 131)

6.4.2.25 Export/Import von Einstellungen für ProDiag-Überwachungen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie den Export/Import von Überwachungseinstellungen unterstützen.

Unter `SupervisionSettingsExportImportResult` sind die folgenden Eigenschaften verfügbar:

Name der Eigenschaft	Datentyp	Zugriff
State	<code>SupervisionSettingExportImportResultState</code>	Lesen
ErrorCount	<code>Int32</code>	Lesen
WarningCount	<code>Int32</code>	Lesen
Message	<code>String</code>	Lesen

Die Enum `SupervisionSettingExportImportResultState` hat folgende Werte:

ENUM	Wert
<code>SupervisionSettingExportImportResultState</code>	Success
	ErrorRollback

Programmcode

Ändern Sie folgenden Programmcode, um Überwachungseinstellungen zu exportieren:

```
// MyProject obtains SupervisionSettingsProvider
SupervisionSettingsProvider settingsProvider =
MyProject.GetService<SupervisionSettingsProvider>();
// Import file path defined
FileInfo exportFile = new FileInfo(@"D:\Temp\ProDiagSettings.dat");
// Stores result after import
SupervisionSettingsExportImportResult exportResult = settingsProvider.Export(exportFile);
// Result state after import
SupervisionSettingsExportImportResultState resultState = exportResult.State;
// Total error count of result messages
int totalErrorCount = exportResult.ErrorCount;
// Composition of result messages
SupervisionSettingsExportImportResultMessageComposition exportMessageList =
exportResult.Messages;
// Count of result messages
exportMessageList.Count;
// Read out specific message text from each message
exportMessageList [0].Message;
// Output: "Export of file 'D:\temp\SupervisionSettings.dat' with the ProDiag supervision
settings was successful.";
```

Ändern Sie folgenden Programmcode, um Überwachungseinstellungen zu importieren:

```
// MyProject obtains SupervisionSettingsProvider
SupervisionSettingsProvider settingsProvider =
MyProject.GetService<SupervisionSettingsProvider>();
// Import file path defined
FileInfo importFile = new FileInfo(@"D:\Temp\ProDiagSettings.dat");
// Stores result after import
SupervisionSettingsExportImportResult importResult = settingsProvider.Import(importFile);
// Result state after import
SupervisionSettingsExportImportResultState resultState = importResult.State;
// Total error count of result messages
int totalErrorCount = importResult.ErrorCount;
// Composition of result messages
SupervisionSettingsExportImportResultMessageComposition importMessageList =
importResult.Messages;
// Count of result messages
importMessageList.Count;
// Read out specific message text from each message
importMessageList[0].Message;
// Output: "Import of the file 'D:\temp\SupervisionSettings.dat' with the ProDiag
supervision settings was successful.";
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.4.2.26 Beobachtungs- und Forcetabelle exportieren/importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Sie haben mit Ihrer Anwendung TIA Portal Openness ein Projekt geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Mit TIA Portal Openness können Sie die Beobachtungstabelle und die Forcetabelle aus dem TIA Portal in SIMATIC ML exportieren und anschließend die Beobachtungstabelle und die Forcetabelle aus SIMATIC ML importieren.

In der Beobachtungstabelle kann die Export-Option mit folgender Definition eingestellt werden (None, WithDefaults, WithReadOnly, WithDefaultsAndReadOnly). Eine Beobachtungstabelle hat nur eine veröffentlichte Eigenschaft, den Namen. Dieser ist zum Lesen veröffentlicht.

Die importierte Beobachtungstabelle wird der Liste der Beobachtungstabellen hinzugefügt. In der Beobachtungstabelle muss unter den Importoptionen die erforderliche Option

angegeben werden (None, Override). Die Forcetabellen können auf ähnliche Weise importiert werden, es ist jedoch nur genau eine Forcetabelle zulässig.

Wenn Sie statt Override die Option None verwenden und bereits eine (nicht leere) Forcetabelle vorhanden ist, schlägt der Import mit der Ausnahme `RecoverableException` fehl: Es darf nur eine Forcetabelle importiert werden.

Programmcode: Beobachtungstabelle und Forcetabelle exportieren

Sie können den folgenden Programmcode ändern, um die Beobachtungstabelle zu exportieren:

```
SoftwareContainer softwareContainer =  
(IEngineeringServiceProvider)item.GetService<SoftwareContainer>();  
PlcSoftware plcSoftware = softwareContainer.Software as PlcSoftware;  
PlcWatchTableComposition exportWatchTables =  
plcSoftware.PlcWatchAndForceTableGroup.WatchTables;  
PlcWatchTable watchTable = exportWatchTables.Find(watchTableName);  
if(watchTable != null)  
{  
    watchTable.Export((FileInfo) fileInfo, ExportOptions.None);  
}
```

Hinweis

In der Beobachtungstabelle müssen Exportoptionen eingestellt werden (None, WithDefaults, WithReadOnly, WithDefaultsAndReadOnly). Eine Beobachtungstabelle hat nur eine veröffentlichte Eigenschaft, den Namen. Der ist zum Lesen veröffentlicht.

Sie können den folgenden Programmcode ändern, um die Forcetabelle zu exportieren:

```
SoftwareContainer softwareContainer =  
(IEngineeringServiceProvider)item.GetService<SoftwareContainer>();  
PlcSoftware plcSoftware = softwareContainer.Software as PlcSoftware;  
PlcForceTableComposition exportForceTables =  
plcSoftware.PlcWatchAndForceTableGroup.ForceTables;  
PlcForceTable forceTable = exportForceTables[0];  
forceTable.Export((FileInfo) fileInfo, ExportOptions.None);
```

Hinweis

In jeder Situation gibt es nur eine Forcetabelle, deren Name schreibgeschützt ist.

Programmcode: Beobachtungstabelle und Forcetabelle importieren

Sie können den folgenden Programmcode ändern, um die Beobachtungstabelle zu importieren:

```
SoftwareContainer softwareContainer =
  ((IEngineeringServiceProvider)item).GetService<SoftwareContainer>();
PlcSoftware plcSoftware = softwareContainer.Software as PlcSoftware;
PlcWatchTableComposition importWatchTables =
  plcSoftware.PlcWatchAndForceTableGroup.WatchTables;
IList<PlcWatchTable> WatchTables = importWatchTables.Import((FileInfo) fileInfo,
  ImportOptions.None);
```

Hinweis

Die importierte Beobachtungstabelle wird der Liste der Beobachtungstabellen hinzugefügt. In der Beobachtungstabelle müssen die erforderlichen Importoptionen eingestellt werden (None, Override).

Forcetabellen können auf ähnliche Weise importiert werden, es ist jedoch nur genau eine Forcetabelle zulässig. Wenn Sie statt Override die Option None verwenden und bereits eine (nicht leere) Forcetabelle vorhanden ist, schlägt der Import mit der folgenden Ausnahme `RecoverableException: fehl: Es darf nur eine Forcetabelle importiert werden. Verwenden Sie die Importoption Override, um die vorhandene Tabelle zu überschreiben.`

6.4.2.27 Anwenderdatentyp exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Programmcode

Ändern Sie den folgenden Programmcode, um einen Anwenderdatentyp in eine XML-Datei zu exportieren:

```
//Exports a user defined type
private static void ExportUserDefinedType(PlcSoftware plcSoftware)
{
    string udtname = "udt name XYZ";
    PlcTypeComposition types = plcSoftware.TypeGroup.Types;
    PlcType udt = types.Find(udtname);
    udt.Export(new FileInfo(string.Format(@"C:\OpennessSamples\udts\{0}.xml", udt.Name)),
    ExportOptions.WithDefaults);
}
```

6.4.2.28 Anwenderdatentyp importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Einleitung

Die API-Schnittstelle unterstützt das Importieren von Anwenderdatentypen aus einer XML-Datei.

Syntax der Importdatei

Der folgende Beispielcode zeigt einen Ausschnitt aus einer Importdatei eines benutzerdefinierten Datentyps:

```
<Section Name="Input">
  <Member Name="Input1" Datatype="myudt1">
    <Sections>
      <Section Name="None">
        <Member Name="MyUDT1Member1" Datatype="bool"/>
        <Member Name="MyUDT1Member2" Datatype="myudt1">
          <Sections...
        </Member>
      </Section>
    </Sections>
  </Member>
</Section>
```

Hinweis

Syntax für benutzerdefinierte Datentypen von Elementen

Wenn der benutzerdefinierte Datentyp eines Elements in der Importdatei für Anwenderdatentypen eine falsche Syntax aufweist, wird eine Ausnahme ausgelöst.

Achten Sie darauf, dass benutzerdefinierte Datentypen mit `"` notiert werden.

Programmcode

Um einen Anwenderdatentyp zu importieren, ändern Sie folgenden Programmcode:

```
//Imports user data type
private static void ImportUserDataTypes(PlcSoftware plcSoftware)
{
    FileInfo fullFilePath = new FileInfo(@"C:\OpennessSamples\Import\ExportedPlcType.xml");
    PlcTypeComposition types = plcSoftware.TypeGroup.Types;
    IList<PlcType> importedTypes = types.Import(fullFilePath, ImportOptions.Override);
}
```

Siehe auch

Import von Projektierungsdaten (Seite 1255)

6.4.2.29 Export von Daten im Format OPC UA XML

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist nicht online.

Verwendung

Sie können mit TIA Portal Openness PLC-Daten als Datei im Format OPC UA XML exportieren. Als Eingabeparameter für die Aktion benötigen Sie einen absoluten Verzeichnispfad, in dem die XML-Datei gespeichert wird.

Programmcode

Ändern Sie folgenden Programmcode, um PLC-Daten als OPC UA XML-Datei zu exportieren:

```
//Export PLC data as OPC UA XML file
private static void OpcUaExport(Project project, DeviceItem plc)
{
    OpcUaExportProvider opcUaExportProvider = project.HwUtilities.Find("OPCUAExportProvider")
    as OpcUaExportProvider;
    if (opcUaExportProvider == null) return;
    opcUaExportProvider.Export(plc, new FileInfo(string.Format(@"D:\OPC UA export
files\{0}.xml", plc.Name)));
}
```

6.4.2.30 Export/Import von UDT & DB

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Sie können TIA Portal Openness verwenden, um eine Überwachung für boole'sche Mitglieder in einem UDT festzulegen und ProDiagFB für UDT-Instanzen in den globalen Daten zuzuweisen, wenn der globale Datenbaustein erstellt wird. Sie sollten in der Lage sein, diese Überwachungsinformation über den Export bzw. Import in TIA Portal Openness bereitzustellen bzw. abzurufen.

XML-Struktur exportierter/importierter UDT

Verwenden Sie die folgende XML-Struktur, um ein Element einer boole'schen Variablen, für die Sie eine Überwachung festgelegt haben, zu exportieren:

6.4 Daten eines PLC-Geräts importieren/exportieren

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
<Engineering version="V17" />
<DocumentInfo>
<Created>2020-06-08T20:10:12.6308242Z</Created>
<ExportSetting>None</ExportSetting>
<InstalledProducts>
<Product>
<DisplayName>Totally Integrated Automation Portal</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<OptionPackage>
<DisplayName>TIA Portal Openness</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</OptionPackage>
<OptionPackage>
<DisplayName>TIA Portal Version Control Interface</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</OptionPackage>
<Product>
<DisplayName>Feature Cycle 3 TIA Portal</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<Product>
<DisplayName>STEP 7 Professional</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<OptionPackage>
<DisplayName>SIMATIC Energy Suite</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</OptionPackage>
<OptionPackage>
<DisplayName>STEP 7 Safety</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</OptionPackage>
<Product>
<DisplayName>WinCC Professional</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
</InstalledProducts>
</DocumentInfo>
<SW.Types.PlcStruct ID="0">
<AttributeList>
<Interface><Sections>
<Section Name="None">
<Member Name="Element_1" Datatype="Bool" />
</Section>
</Sections>
</Interface>
<Name>User_data_type_1</Name>
<Supervisions><PLCDataTypeSupervisions>
<PLCDataTypeSupervision Number="1" Type="Operand">
<SupervisedOperand Name="#Element_1" />
<SupervisedStatus>false</SupervisedStatus>
<DelayOperand Name="T#0ms" />

```

```

<Conditions>
<Condition>
<ConditionOperand Number="1" Name="" />
<TriggeringStatus>true</TriggeringStatus>
</Condition>
<Condition>
<ConditionOperand Number="2" Name="" />
<TriggeringStatus>true</TriggeringStatus>
</Condition>
<Condition>
<ConditionOperand Number="3" Name="" />
<TriggeringStatus>true</TriggeringStatus>
</Condition>
</Conditions>
<CategoryNumber>1</CategoryNumber>
<SubCategory1Number>0</SubCategory1Number>
<SubCategory2Number>0</SubCategory2Number>
</PLCDataTypeSupervision></PLCDataTypeSupervisions></Supervisions>
</AttributeList>
<ObjectList>
<MultilingualText ID="1" CompositionName="Comment">
<ObjectList>
<MultilingualTextItem ID="2" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
<MultilingualText ID="3" CompositionName="Title">
<ObjectList>
<MultilingualTextItem ID="4" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
</ObjectList>
</SW.Types.PlcStruct>
</Document>

```

Wenn Sie einen UDT importieren und eine Überwachung für die boole'sche Variable zuweisen wollen, können obige XML-Variablen verwendet werden. Sie können jedoch nach dem Import keine Überwachung für eine boole'sche Variable über Openness einrichten.

Hinweis

Die Überwachung ist nur für boole'sche Variablen für UDT möglich; wenn Sie versuchen, die Variablen "PLCDataTypeSupervisions" für nicht-boole'sche Mitglieder in UDT zu importieren oder zu exportieren, wird keine Ausnahme erzeugt.

XML-Struktur exportierter/importierter DB

Verwenden Sie die folgende XML-Struktur, um ein Mitglied einer UDT-Instanz zu exportieren, unabhängig davon ob der UDT mit oder ohne Überwachung eingerichtet ist und ob ein Struct-Array mit UDT-Instanzen als Mitglied vorhanden ist:

```

<?xml version="1.0" encoding="utf-8"?>
<Document>
<Engineering version="V17" />
<DocumentInfo>
<Created>2020-06-09T16:01:18.494539Z</Created>
<ExportSetting>None</ExportSetting>
<InstalledProducts>
<Product>
<DisplayName>Totally Integrated Automation Portal</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<Product>
<DisplayName>Feature Cycle 1 TIA Portal</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<Product>
<DisplayName>Feature Cycle 3 TIA Portal</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<Product>
<DisplayName>STEP 7 Professional</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<OptionPackage>
<DisplayName>STEP 7 Safety</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</OptionPackage>
</InstalledProducts>
</DocumentInfo>
<SW.Blocks.GlobalDB ID="0">
<AttributeList>
<Interface><Sections>
<Section Name="Static">
<Member Name="Static_1" Datatype="Udt_With_Supervision"><AttributeList><BooleanAttribute
Name="SetPoint" SystemDefined="true">true</BooleanAttribute></
AttributeList><AssignedProDiagFB>Default_SupervisionFB</AssignedProDiagFB></Member>
<Member Name="Static_2" Datatype="Array[0..1] of Struct">
<Member Name="Static_1"
Datatype="&quot;Udt_With_Supervision&quot;"><AttributeList><BooleanAttribute
Name="SetPoint" SystemDefined="true">true</BooleanAttribute></AttributeList><Subelement
Path="0"><AssignedProDiagFB>Default_SupervisionFB</AssignedProDiagFB></
Subelement><Subelement Path="1"><AssignedProDiagFB>Default_SupervisionFB</
AssignedProDiagFB></Subelement></Member>
</Member>
</Section>
</Sections></Interface>
<MemoryLayout>Optimized</MemoryLayout>
<MemoryReserve>100</MemoryReserve>
<Name>Data_block_5</Name>
<Number>16</Number>
<ProgrammingLanguage>DB</ProgrammingLanguage>
</AttributeList>
<ObjectList>
<MultilingualText ID="1" CompositionName="Comment">

```

```
<ObjectList>
<MultilingualTextItem ID="2" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
<MultilingualText ID="3" CompositionName="Title">
<ObjectList>
<MultilingualTextItem ID="4" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
</ObjectList>
</SW.Blocks.GlobalDB>
</Document>
```

Hinweis

Attribut "AssignedProDiagFB" ist nur für UDT-Instanzen gültig. Es ist nicht möglich, ungültige Daten zu exportieren, da sie sauber übersetzt werden sollten und daher keine Ausnahme während des Exports erzeugt wird. Sie werden die Ausnahme "Not supported" erhalten, wenn Sie versuchen, Variable "AssignedProDiagFB" für nicht-UDT-Instanzen im globalen DB oder in ein anderes Mitglied in einem beliebigen Baustein zu importieren.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)
Projekt öffnen (Seite 131)

6.4.2.31 Export/Import von Array- & Instanz-DB

Zusammenfassung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können mit TIA Portal Openness den Array-DB und Instanz-DB von UDT-Instanzen exportieren und importieren, sodass die Informationen über den zugewiesenen ProDiagFB korrekt exportiert werden sollten.

Ebenso sollte das gleiche Attribut für Openness-Nutzer auch beim Import unterstützt werden, sodass Sie den möglichen ProDiagFB für den Array-DB und Instanz-DB von UDT zuweisen können.

XML-Struktur des exportierten Array-DB

Verwenden Sie die folgende XML-Struktur für den Export eines Array-DB eines UDT, dem ProDiagFB "Default_SupervisionFB" zugewiesen ist:


```

<?xml version="1.0" encoding="utf-8"?>
<Document>
<Engineering version="V17" />
<DocumentInfo>
<Created>2020-06-09T16:06:07.7850963Z</Created>
<ExportSetting>None</ExportSetting>
<InstalledProducts>
<Product>
<DisplayName>Totally Integrated Automation Portal</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<Product>
<DisplayName>Feature Cycle 1 TIA Portal</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<Product>
<DisplayName>Feature Cycle 3 TIA Portal</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<Product>
<DisplayName>STEP 7 Professional</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</Product>
<OptionPackage>
<DisplayName>STEP 7 Safety</DisplayName>
<DisplayVersion>V17</DisplayVersion>
</OptionPackage>
</InstalledProducts>
</DocumentInfo>
<SW.Blocks.ArrayDB ID="0">
<AttributeList>
<Interface><Sections>
<Section Name="None">
<Member Name="Data_block_6" Datatype="Array[0..1] of &quot;Udt_With_Supervision&quot;">
<Comment>
<MultiLanguageText Lang="en-US">comment of Data_block_6</MultiLanguageText>
</Comment>
<Sections>
<Section Name="None">
<Member Name="Element_1" Datatype="Bool">
<Subelement Path="0">
<Comment>
<MultiLanguageText Lang="en-US">comment of Element_1</MultiLanguageText>
</Comment>
</Subelement>
<Subelement Path="1">
<Comment>
<MultiLanguageText Lang="en-US">comment of Element_1</MultiLanguageText>
</Comment>
</Subelement>
</Member>
</Section>
</Sections>
<Subelement Path="0">
<Comment>

```

6.4 Daten eines PLC-Geräts importieren/exportieren

```
<MultiLanguageText Lang="en-US">comment of Data_block_6[0]</MultiLanguageText>
</Comment>
<AssignedProDiagFB>Default_SupervisionFB</AssignedProDiagFB>
</Subelement>
<Subelement Path="1">
<Comment>
<MultiLanguageText Lang="en-US">comment of Data_block_6[1]</MultiLanguageText>
</Comment>
<AssignedProDiagFB>Default_SupervisionFB</AssignedProDiagFB>
</Subelement>
</Member>
</Section>
</Sections></Interface>
<Name>Data_block_6</Name>
<Number>17</Number>
<ProgrammingLanguage>DB</ProgrammingLanguage>
</AttributeList>
<ObjectList>
<MultilingualText ID="1" CompositionName="Comment">
<ObjectList>
<MultilingualTextItem ID="2" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
<MultilingualText ID="3" CompositionName="Title">
<ObjectList>
<MultilingualTextItem ID="4" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
</ObjectList>
</SW.Blocks.ArrayDB>
</Document>
```

Hinweis

Es ist möglich, AssignedProDiagFB nur für den Array-DB eines UDT zu setzen, der Export ungültiger Daten ist nicht möglich, sodass Sie beim Export keine Ausnahme erhalten.

Wenn Sie versuchen, Variable "AssignedProDiagFB" für einen anderen Array-DB zu importieren, der kein UDT ist, wird Ausnahme "Not Supported" erzeugt und dies wird auch nicht für systemdefinierte Datentypen unterstützt, unabhängig davon, ob diese boole'sche Variablen haben oder nicht.

Exportierte XML-Struktur eines Instanz-DB

Verwenden Sie die folgende XML-Struktur für den Export eines Instanz-DB eines UDT, dem ProDiagFB "Default_SupervisionFB" zugewiesen ist:

```
...
<SW.Blocks.InstanceDB ID="0">
<AttributeList>
<AssignedProDiagFB>Default_SupervisionDB</AssignedProDiagFB>
<InstanceOfName>User_data_type_4</InstanceOfName>
<InstanceOfType>UDT</InstanceOfType>
<Interface>
<Sections>
<Section Name="Static">
<Member Name="Element_1" Datatype="Bool" />
</Section>
</Sections>
</Interface>
<MemoryLayout>Optimized</MemoryLayout>
<Name>Data_block_9</Name>
<Number>19</Number>
<ProgrammingLanguage>DB</ProgrammingLanguage>
</AttributeList>
<ObjectList>
<MultilingualText ID="1" CompositionName="Comment">
<ObjectList>
<MultilingualTextItem ID="2" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
<MultilingualText ID="3" CompositionName="Title">
<ObjectList>
<MultilingualTextItem ID="4" CompositionName="Items">
<AttributeList>
<Culture>en-US</Culture>
<Text />
</AttributeList>
</MultilingualTextItem>
</ObjectList>
</MultilingualText>
</ObjectList>
</SW.Blocks.InstanceDB>
</Document>
...
```

Hinweis

Die Ausnahme "Not Supported attribute" sollte erzeugt werden, wenn Sie versuchen, AssignedProDiagFB im Instanz-DB zu setzen.

Siehe auch

Projekt öffnen (Seite 131)

6.4.3 Technologieobjekte**6.4.3.1 Übersicht über Technologieobjekte und Versionen****Technologieobjekte**

Die folgende Tabelle zeigt die verfügbaren Technologieobjekte für den Import und Export.

CPU	Technologie	Technologieobjekt	Version des Technologieobjekts	CPU FW
S7-1200	Motion Control	TO_PositioningAxis	≥ V7.0	≥ V4.4
		TO_CommandTable		
	PID-Regelung	PID_Compact	≥ V2.3	≥ V4.2
		PID_3Step	V2.3	
		PID_Temp	V1.1	

CPU	Technologie	Technologieobjekt	Version des Technologieobjekts	CPU FW	
S7-1500	Motion Control	TO_SpeedAxis	≥ V5.0	≥ V2.8	
		TO_PositioningAxis			
		TO_ExternalEncoder			
		TO_SynchronousAxis			
		TO_OutputCam			
		TO_CamTrack			
		TO_MeasuringInput			
		TO_Cam (S7-1500T)			
		TO_Kinematics (S7-1500T)			
		TO_LeadingAxisProxy (S7-1500T)			
		TO_Cam_10k (S7-1500T)	≥ V6.0	≥ V2.9	
		TO_Interpreter (S7-1500T)	≥ V8.0	≥ V3.1	
		TO_InterpreterMapping (S7-1500T)			
		TO_InterpreterProgram (S7-1500T)			
	Zählen und Messen	High_Speed_Counter	≥ V4.1	Jede	
		SSI_Absolute_Encoder	≥ V3.1		
	PID-Regelung	PID-Regelung	PID_Compact	≥ V2.3	≥ V2.0
			PID_3Step	V2.3	
			PID_Temp	V1.1	
			CONT_C	V1.1	Jede
			CONT_S		
			TCONT_CP		
			TCONT_S		
S7-300/400			PID-Regelung	CONT_C	V1.1
	CONT_S				
	TCONT_CP				
	TCONT_S				
	TUN_EC				
	TUN_ES				
	PID_CP	V2.0			
	PID_ES				
	EMC	AXIS_REF	V2.0		

6.4.3.2 XML-Struktur des Schnittstellenabschnitts Technologieobjekt

Grundprinzip

Die Daten in der XML-Datei vom Import/Export sind mit Verweis auf ein Openness-XML-Dateiformat gegliedert. Jede Importdatei muss die strukturellen Bedingungen erfüllen.

Die Exportdatei enthält alle bearbeiteten Variablen und Konstanten des Schnittstellenabschnitts eines exportierten Technologieobjekt-Datenbausteins.

Die Projektdaten können mehr Daten als die Import-XML-Datei enthalten, z.B. externe Referenzen. Diese müssen separat exportiert werden.

Nur schreibbare Werte können über TIA Portal Openness XML importiert werden.

Abhängig von den Exporteinstellungen in TIA Portal Openness enthält die Exportdatei einen festgelegten Satz von Attributen und Elementen. Exportdateien aus höheren Versionen des TIA Portals sind nicht mit früheren Versionen des TIA Portals kompatibel und können nicht in diese Versionen importiert werden. SimaticML-Dateien, die mit der neuesten TechObject-Version exportiert wurden, werden auch für die neueren Versionen unterstützt.

Grundstruktur

Das SimaticML-Format entspricht beinahe dem Export/Import-Format der Anwenderbausteine.

Bei Technologieobjekten sind im Element SW.TechnologicalObject.TechnologicalInstanceDB die folgenden Elemente erforderlich.

```
<SW.TechnologicalObjects.TechnologicalInstanceDB ID="0">
  <AttributeList>
    <InstanceOfName>TO_SpeedAxis</InstanceOfName>
    <Interface>
      ...
    </Interface>
    <Name>SpeedAxis_1</Name>
    <Number>1</Number>
    <OfSystemLibElement>TO_SpeedAxis</OfSystemLibElement>
    <OfSystemLibVersion>8.0</OfSystemLibVersion>
    <ParameterData>
      ...
    </ParameterData>
    <ProgrammingLanguage>Motion_DB</ProgrammingLanguage>
  </AttributeList>
  <ObjectList>
    ...
  </ObjectList>
</SW.TechnologicalObjects.TechnologicalInstanceDB>
```

- InstanceOfName
Name der Instanz, von der das Technologieobjekt abgeleitet wird.
- Interface
 - Sections-Elemente stellen die Hierarchiestruktur der Technologieobjekt-Datentypen und deren Version dar.
 - Member-Elemente stellen alle Parameter dar, die direkt auf die Technologieobjekt-Datenbausteinvariablen abgebildet sind.
 - Die AttributeList umfasst alle definierten Attribute eines Mitglieds. Attribute, die vom System festgelegt oder von einem Default-Wert zugewiesen sind, werden in der XML-Struktur nicht aufgeführt. Die Mitgliedsattribute <ReadOnly> und <Informative> werden nur in die XML-Exportdatei geschrieben, wenn ihr Wert TRUE ist.
- Name
Name des Technologieobjekts im TIA Portal-Projekt
- Number
Nummer des Technologieobjekt-Datenbausteins im TIA Portal-Projekt
- OfSystemLibElement
Typ des Technologieobjekts
- OfSystemLibVersion
Version des Technologieobjekts
- ParameterData
 - Parameter-Elemente stellen Parameter dar, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen oder Parameter abgebildet sind, bei denen es sich um schreibgeschützte Technologieobjekt-Datenbausteinvariablen handelt, die jedoch in der PublicAPI schreibbar sind.
 - AdditionalData-Elemente sind Informationen, die nicht in Datenbausteinvariablen für Technologieobjekte gespeichert werden, z.B. Verschaltungen mit Antrieben oder anderen Technologieobjekten.
- ProgrammingLanguage
Der einzige gültige Inhalt ist Motion_DB.

Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Die Reihenfolge der folgenden Beschreibung von Elementen stellt die Parameter eines direkt auf Technologieobjekt-Datenbausteine abgebildeten Technologieobjekts Drehzahlachse dar.

```
<Section Name="Static">
...
  <Member Name="DynamicLimits" Datatype="TO_Struct_DynamicLimits" Version="6.0">
    <AttributeList>
      <BooleanAttribute Name="SetPoint" SystemDefined="true">true</BooleanAttribute>
    </AttributeList>
    <Sections>
      <Section Name="None">
        <Member Name="MaxAcceleration" Datatype="LReal">
          <StartValue>1000.0</StartValue>
        </Member>
        <Member Name="MaxDeceleration" Datatype="LReal">
          <StartValue>1000.0</StartValue>
        </Member>
        <Member Name="MaxJerk" Datatype="LReal">
          <StartValue>20000.0</StartValue>
        </Member>
      </Section>
    </Sections>
  </Member>
...
</Section>
```

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind

Die Reihenfolge der folgenden Beschreibung von Elementen stellt die Parameter eines nicht direkt auf Technologieobjekt-Datenbausteine abgebildeten Technologieobjekts Drehzahlachse dar.

```
<ParameterData>
  <Parameters xmlns="http://www.siemens.com/automation/Openness/SW/Parameters/v1">
    <Parameter Name="_Units.VelocityUnit" />
    <Parameter Name="_Units.TorqueUnit" />
    <Parameter Name="Actor.Type" />
    <Parameter Name="Actor.Interface.EnableTorqueData" />
    <Parameter Name="Actor.Interface.EnableDriveOutput" />
    <Parameter Name="Actor.Interface.DriveReadyInput" />
    <Parameter Name="_Actor.Interface.EnableDriveOutputAddress" />
    <Parameter Name="_Actor.Interface.DriveReadyInputAddress" />
    <Parameter Name="_Actor.Interface.Telegram" />
    <Parameter Name="_Actor.DataAdaptionOffline" />
    <AdditionalData xmlns="http://www.siemens.com/automation/Openness/SW/Motion/Axis/v1/">
      <Connection Interface="Actor" OutputTag="myDrive1" />
      <Connection Interface="Torque" />
    </AdditionalData>
  </Parameters>
</ParameterData>
```

Weitere Informationen

Weitere Informationen zum Interface-Element und zu Attributen der SimaticML-Elemente finden Sie unter XML-Struktur des Abschnitts Bausteinschnittstelle (Seite 1338).

6.4.3.3 Technologieobjekte exportieren

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Ein Projekt enthält das zu exportierende TO.
- Die PLC ist nicht online.

Verwendung

Die TIA Portal Openness API unterstützt den Export aller Technologieobjekte in der Tabelle Übersicht über Technologieobjekte und Versionen (Seite 1448) in eine XML-Datei. Sie können nur konsistente Technologieobjekte exportieren. Die Hardwarekonfiguration oder Variablentabellen werden nicht zusammen mit dem Technologieobjekt exportiert, sondern müssen separat exportiert werden. Sie können die Konsistenz eines Technologieobjekts mit dem Attribut IsConsistent prüfen. Dieser Merker kann bei erfolgreichem Übersetzen des jeweiligen Technologieobjekts gesetzt werden. Die Generierung der entsprechenden Exportdatei weist darauf hin, dass der Export abgeschlossen ist.

In den folgenden Fällen wird eine Ausnahme erzeugt:

- Beim Versuch, ein inkonsistentes Technologieobjekt zu exportieren.
- Beim Versuch, ein Technologieobjekt oder die Version eines Technologieobjekts zu exportieren, das die Exportfunktionalität nicht unterstützt.
- Beim Versuch, ein Technologieobjekt zu exportieren, während Sie im Online-Modus sind.

Parameter von Motion Control-Technologieobjekten

Die meisten Parameter von Motion Control-Technologieobjekten sind direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet. Diese Parameters werden in das XML-Element Member im Abschnitt Interface exportiert.

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind, werden in das XML-Element ParameterData exportiert.

Detaillierte Beschreibung der Parameter:

- Spezifische Parameter für TO_PositioningAxis und TO_CommandTable (Seite 1456),
- Spezifische Parameter für die Technologieobjekte Achse und Geber (Seite 1459),
- Spezifische Parameter für Technologieobjekte Leitachsstellvertreter (Seite 1462)
- Spezifische Parameter für das Technologieobjekt Messtaster (Seite 1462)
- Spezifische Parameter für die Technologieobjekte Nocken und Nockenspur (Seite 1463)
- Spezifische Parameter für Technologieobjekte Kurvenscheibe (Seite 1464)
- Spezifische Parameter für Technologieobjekte Kinematik (Seite 1468)

Programmcode

```
void Export(FileInfo path, ExportOptions options);
```

Der Parameter path beschreibt den Pfad und den Dateinamen der Exportdatei, die erstellt werden soll. Der Parameter (ExportOptions options) gibt Optionen für den Export (Seite 1253) an.

Attribute	Beschreibung
ExportOptions.None	Exportiert nur geänderte Parameter.
ExportOptions.WithDefaults	Diese Option exportiert die Parameter, die ihren Standardwert haben. Der Standardwert selbst wird nicht exportiert. Das gleiche Verhalten ist auch für die TO-spezifischen Teile des XML-Exports definiert. Die jeweiligen Standardwerte, die exportiert werden, sind für jeden TO-Typ festgelegt.
ExportOptions.WithReadOnly	Diese Option exportiert zusätzliche informative Attribute und Werte. Beim Import werden schreibgeschützte Werte nicht in den DB zurückgeschrieben. Die jeweiligen schreibgeschützten Werte, die exportiert werden, sind für jeden TO-Typ definiert.
ExportOptions.WithDefaults ExportOptions.With-ReadOnly	Kombination aus den zwei Optionen oben.

Um ein Technologieobjekt zu suchen und in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
// Find a specific technology object by its name and export this
private static void ExportTechnologicalObject(FileInfo path, ExportOptions options,
PlcSoftware plcSoftware, String nameOfTO)
{
    // Find by name
    TechnologicalInstanceDBComposition technologicalObjects = plcSoftware.TechnologicalObje
ctGroup.TechnologicalObjects;
    TechnologicalInstanceDB technologicalObject = technologicalObjects.Find(nameOfTO);
    // Export TO
    technologicalObject.Export(path, options);
}
```

Um ein Technologieobjekt OutputCam in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
// Export OutputCam
private static void ExportOutputCam(FileInfo path, ExportOptions options,
TechnologicalInstanceDB parentTO, String nameOfOutputCam)
{
    //Retrieve service OutputCamMeasuringInputContainer
    OutputCamMeasuringInputContainer container =
    parentTO.GetService<OutputCamMeasuringInputContainer>();
    //Get access to TO_OutputCam container
    TechnologicalInstanceDBComposition outputcamCamtrackContainer = container.OutputCams;
    //Find technology object TO_OutputCam
    TechnologicalInstanceDB outputCam = outputcamCamtrackContainer.Find("nameOfOutputCam");
    // Export
    outputCam.Export(path, options);
}
```

6.4.3.4 Technologieobjekte importieren

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist nicht online.

Verwendung

Die TIA Portal Openness API unterstützt den Import von Technologieobjekten aus einer XML-Datei.

Ausnahmen

Wenn die Importdaten Parameter enthalten, die für den jeweiligen TO-Typ nicht definiert sind, wird eine `EngineeringTargetException` gemeldet.

Wenn der Wert eines Parameters ein falsches Format hat und nicht in den Typ des Parameters konvertiert werden kann, wird eine `EngineeringTargetException` gemeldet. Der Import wird durchgeführt, doch beim Übersetzen wird ein Fehler erzeugt.

Offene Verbindung

Im TIA Portal werden offene Verbindungen erstellt, wenn eine Variable, die mit einem TO verschaltet ist, gelöscht wird. Das weitere Verhalten offener Verbindungen ist identisch mit dem Verhalten offener Verbindungen, die durch Löschen einer verschalteten Variablen erstellt wurden. Wenn das Partner-TO beim Import fehlt, wird eine offene Verbindung erstellt. Das gilt auch für TO-Typen, die für bestimmte Verbindungen nicht verwendet werden können.

Programmcode

```
ICollection<SW.TechnologicalObjects.TechnologicalInstanceDB>  
Import(FileInfo path, ImportOptions options);
```

Um eines oder mehrere Technologieobjekte aus einer XML-Datei zu importieren, ändern Sie den folgenden Programmcode. Weitere Informationen zu den ImportOptions finden Sie unter Import von Projektierungsdaten (Seite 1255).

```
// Import technology objects  
private static void Import(FileInfo path, ImportOptions options, PlcSoftware plcSoftware)  
{  
    // Create technological instance  
    TechnologicalInstanceDBComposition technologicalObjects =  
    plcSoftware.TechnologicalObjectGroup.TechnologicalObjects;  
  
    // Import TO  
    technologicalObjects.Import(path, options);  
}
```

6.4.3.5 S7-1200 Motion Control

Spezifische Parameter für TO_PositioningAxis und TO_CommandTable

Verwendung

Die API unterstützt den Export und Import von Technologieobjekten.

Parameter von TO_PositioningAxis und TO_CommandTable, bei denen es sich um Datenbausteinmitglieder handelt, werden in den Abschnitt "Interface" der Exportdatei exportiert. Zusätzliche Parameter werden in den Abschnitt "ParameterData" der Exportdatei exportiert. Das nächste Kapitel zeigt alle zusätzlichen Parameter.

Eine Liste mit allen verfügbaren Variablen finden Sie im Funktionshandbuch SIMATIC STEP 7 S7-1200 Motion Control im Internet (<https://support.industry.siemens.com/cs/ww/de/view/109773400>).

Hinweis

Die Datentypinformation wird nur für Parameter exportiert, die Bestandteil des Abschnitts "Interface" sind.

Zusätzliche Parameter TO_PositioningAxis

Die folgenden zusätzlichen Parameter sind DB-Mitgliedern nicht direkt zugeordnet und werden im Abschnitt ParameterData abhängig von der ExportOption exportiert:

Parametername	Mögliche Werte
_Actor.Interface.PTO	Impulsbildnerdaten: 0: Pulse_1 1: Pulse_2 2: Pulse_3 3: Pulse_4
_Actor.Interface.EnableDriveOutput	Variablenname der Adresse des verschalteten Geräts
_Actor.Interface.DriveReadyInput	Variablenname der Adresse des verschalteten Geräts
_Actor.Interface.DataConnection	0: Antrieb 1: DB
_Actor.Interface.DataBlock	Siehe Funktionssicht für mögliche Werte
_Actor.Interface.Analog	Siehe Funktionssicht für mögliche Werte
_Actor.Interface.ProfiDriveIn	Variablenname der Adresse des verschalteten Geräts
_Actor.Interface.ProfiDriveOut	Variablenname der Adresse des verschalteten Geräts
_Actor.DataAdaptionOffline	TRUE oder FALSE
_Sensor[1].Interface.ProfiDriveIn	Variablenname der Adresse des verschalteten Geräts
_Sensor[1].Interface.ProfiDriveOut	Variablenname der Adresse des verschalteten Geräts
_Sensor[1].Interface.EncoderConnection	4: HSC 7: Geber am PROFINET/PROFIBUS
_Sensor[1].DataAdaptionOffline	TRUE oder FALSE
_Sensor[1].Interface.DataConnection	0: Geber 1: DB
_Sensor[1].Interface.HSC	HSC number: 0: HSC_1 1: HSC_2 2: HSC_3 3: HSC_4
_Sensor[1].Interface.DataBlock	Siehe Funktionssicht für mögliche Werte
_Sensor[1].PassiveHoming.DigitalInput	Variablenname der Adresse des verschalteten Geräts
_Sensor[1].ActiveHoming.DigitalInput	Variablenname der Adresse des verschalteten Geräts
_PositionLimits_HW.MinSwitch	Variablenname der Adresse des verschalteten Geräts
_PositionLimits_HW.MaxSwitch	Variablenname der Adresse des verschalteten Geräts

Zusätzliche Parameter TO_CommandTable

Die folgenden zusätzlichen Parameter sind DB-Mitgliedern nicht direkt zugeordnet:

Parametername	Mögliche Werte
_WarningsEnable	TRUE oder FALSE
_UseAxisParametersFrom	"Sample axis", Name der Achse

Externe Referenzen

Externe Referenzen am TO_PositioningAxis

Das Konfigurationsfenster des TO_PositioningAxis zeigt Informationen an, die nicht in Technologieobjekt-Datenbausteinvariablen gespeichert werden. Diese externen Informationen sind somit nicht Bestandteil der exportierten XML-Datei.

Die von der Achse für die Zuordnung zwischen symbolischen Namen und Adressen verwendeten Variablen werden in einer Variablentabelle gespeichert und müssen separat exportiert und importiert werden.

Sofern die folgenden Parameter noch nicht in der Hardwarekonfiguration festgelegt wurden, müssen sie über Parameter eines Technologieobjekts schreiben (Seite 560) eingestellt werden.

Bei PTO-Achse:

Ein expliziter Satz von _Actor.Interface.PTO , um das Technologieobjekt mit dem PTO-Output zu verschalten.

- _Actor.Interface.PTO_OutputA
- _Actor.Interface.PTO_OutputBEnable
- _Actor.Interface.PTO_OutputB
- _Actor.Interface.PTO_SignalType

Für das Referenzieren verwendete Digitaleingänge

- _Sensor[1].ActiveHoming.DigitalInput
- _Sensor[1].PassiveHoming.DigitalInput

Für Positionsgrenzen verwendete Digitaleingänge

- _PositionLimits_HW.MaxSwitch
- _PositionLimits_HW.MinSwitch

Bei Analog/ProfiDrive:

Ein expliziter Satz der folgenden Parameter, um PIP für OB-Servo zu aktivieren:

- _Actor.Interface.ProfiDriveOutName
- _Actor.Interface.ProfiDriveInName
- _Sensor[1].Interface.ProfiDriveOutName
- _Sensor[1].Interface.ProfiDriveInName

Wenn HSC für die Geberschaltung verwendet wird:

Ein expliziter Satz von `_Sensor[1].Interface.HSC`, um das Technologieobjekt mit dem HSC zu verschalten und den HSC in der Hardwarekonfiguration zu aktivieren.

- `_Sensor[1].Interface.HSC_OperatingMode`
- `_Sensor[1].Interface.HSC_InputA`
- `_Sensor[1].Interface.HSC_InputB`

6.4.3.6 S7-1500 Motion Control**Spezifische Parameter für die Technologieobjekte Achse und Geber****Verwendung**

Die API unterstützt den Import und Export von Technologieobjekten. Im folgenden Abschnitt werden die spezifischen Parameter beschrieben.

TO_SpeedAxis, TO_PositioningAxis, TO_SynchronousAxis und TO_ExternalEncoder exportieren

Einige Parameter, die auf schreibgeschützte Technologieobjekt-Datenbausteinvariablen abgebildet sind, sind in der PublicAPI schreibbar. Diese Parameter werden nach ParameterData in der exportierten XML-Datei exportiert. Der Parametername, die zulässigen Werte und die Standardwerte sind die gleichen wie die Werte für die entsprechenden Datenbausteinvariablen.

Die betroffenen Parameter sind in den folgenden Tabellen aufgelistet:

Name	TO_SpeedAxis	TO_PositioningAxis / TO_SynchronousAxis	TO_ExternalEncoder
Actor.Type	X	X	-
Actor.Interface.EnableDriveOutput	X	X	-
Actor.Interface.DriveReadyInput	X	X	-
Actor.Interface.EnableTorqueData	X	X	-
VirtualAxis.Mode	X	X	-
Sensor[n].Existent ¹⁾	-	X	-
Sensor[n].Interface.Number ¹⁾	-	X	-
Sensor[n].Type ¹⁾	-	X	-
Sensor.Interface.Number	-	-	X

6.4 Daten eines PLC-Geräts importieren/exportieren

Sensor.Type	-	-	X
CrossPlcSynchronousOperation.Interface[n].EnableLeadingValueOutput ⁵⁾	-	X	X

¹⁾ S7-1500 CPU: n=1; S7-1500T CPU: 1≤n≤4

⁴⁾ ≥ V5.0

⁵⁾ V5.0 n=1; ≥V6.0 1≤n≤8

Einige Parameter sind nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet. Diese Parameter werden nach ParameterData in der exportierten XML-Datei exportiert. Der Parametername, die zulässigen Werte und die Standardwerte sind die gleichen wie die Werte für die entsprechenden Datenbausteinvariablen.

Die betroffenen Parameter sind in den folgenden Tabellen aufgelistet:

Name	Standardwert	TO_SpeedAxis	TO_PositioningAxis / TO_SynchronousAxis	TO_ExternalEncoder
_Actor.DataAdaptionOffline	false	X	X	-
_Actor.Interface.Telegram	0	X	X	-
_Units.LengthUnit	1013 (mm)	-	X	X
_Units.VelocityUnit	1062 (mm/s)	1083 (1/mm)	X	X
_Units.TorqueUnit	1126 (Nm)	X	X	-
_Units.ForceUnit	1120 (N)	-	X	-
_Sensor[n].DataAdaptionOffline ¹⁾	false	-	X	-
_Sensor[n].Interface.Telegram ¹⁾	0	-	X	-
_Sensor.DataAdaptionOffline	false	-	-	X
_Sensor.Interface.Telegram	false	-	-	X
_Properties.MotionType	0	-	X	X
_Properties.UseHighResolutionPositionValues ⁴⁾	false	-	X	X
_CrossPlcSynchronousOperation.ActivateLocalLeadingValueDelayTimeCalculation ⁴⁾	true	-	X	X

Einige Parameter sind nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet. Diese Parameter werden nach ParameterData in der exportierten XML-Datei exportiert. In SimaticML wird nur der Name der verschalteten Variable exportiert.

Die betroffenen Parameter sind in den folgenden Tabellen aufgelistet:

Name	TO_SpeedAxis	TO_PositioningAxis / TO_SynchronousAxis	TO_ExternalEncoder
_Actor.Interface.EnableDriveOutputAddress	X	X	-
_Actor.Interface.DriveReadyInputAddress	X	X	-
_Sensor[n].ActiveHoming.DigitalInputAddress ¹⁾	-	X	-
_Sensor[n].PassiveHoming.DigitalInputAddress ¹⁾	-	X	-
_Sensor.ActiveHoming.DigitalInputAddress	-	-	X
_Sensor.PassiveHoming.DigitalInputAddress	-	-	X
_PositionLimits_HW.MinSwitchAddress	-	X	-
_PositionLimits_HW.MaxSwitchAddress	-	X	-
_CrossPlcOperation.Interface[n].AddressOut ⁵⁾	-	X	X

XML-Element Connection

Das XML-Element Connection beschreibt die Verschaltungen eines Technologieobjekts. Die Namen und Werte der unten aufgeführten Attribute entsprechen den Parameternamen und Werten der bestimmten AxisEncoderHardwareConnectionInterface.

- Erforderliches Attribut "Interface"
 - Mögliche Werte:

Wert	TO_SpeedAxis	TO_PositioningAxis / TO_SynchronousAxis	TO_ExternalEncoder
Actor	X	X	-
Sensor	-	-	X
Sensor[n] ¹⁾	-	X	-
Torque	X	X	-

1) S7-1500 CPU: n=1; S7-1500T CPU: 1≤n≤4

- Optionale Attribute InputAddress und OutputAddress
 - Beide Attribute müssen entweder zusammen vorhanden sein oder können gar nicht verwendet werden.
 - Die möglichen Werte sind Bitadressen (wie in der API).
 - Das Attribut beschreibt Verbindungen zu DeviceItems und Channels.
- Optionales Attribut ConnectOption
 - Darf nur zusammen mit InputAddress und OutputAddress verwendet werden.
 - Das Attribut entspricht dem Methodenparameter gleichen Namens unter AxisEncoderHardwareConnectionInterface.Connect.
 - Das Attribut hat die möglichen Werte Default und AllowAllModules.

- Optionales Attribut `SensorIndexInActorTelegram`
 - Das Attribut wird für Verbindungen zum Sensorteil im Aktortelegramm verwendet.
 - Die Regeln sind die gleichen wie für das entsprechende API-Attribut definiert.
- Optionales Attribut `PathToDBMember`
 - Das Attribut hat die gleichen möglichen Werte wie für die Parameter der entsprechenden Methode.
 - Beschreibt eine Verbindung zu einem DB-Mitglied.
- Optionales Attribut `OutputTag`
 - Der Wert ist der Name der verschalteten PlcTag.
 - Das Attribut beschreibt eine Verbindung zu einer analogen Variablen.

TO_SynchronousAxis mit Leitwerten verbinden

Sie verbinden `TO_SynchronousAxis` über den Service `SynchronousAxisMasterValues` mit Leitwerten, siehe Gleichlaufachse mit Leitwerten verschalten (Seite 592).

Spezifische Parameter für Technologieobjekte Leitachsstellvertreter

Verwendung

Die API unterstützt den Import und Export von Technologieobjekten. Im folgenden Abschnitt werden die spezifischen Parameter beschrieben.

TO_LeadingAxisProxy exportieren

Der Parameter `_Interface.AddressIn` ist nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet. Dieser Parameter wird nach `ParameterData` in der exportierten XML-Datei exportiert. In SimaticML wird nur der Name der verschalteten Variable exportiert.

Hier gelten die gleichen Regeln wie unter Spezifische Parameter für die Technologieobjekte Achse und Geber (Seite 1459) beschrieben.

Spezifische Parameter für das Technologieobjekt Messtaster

Verwendung

Die API unterstützt den Import und Export von Technologieobjekten. Im folgenden Abschnitt werden die spezifischen Parameter beschrieben.

TO_MeasuringInput exportieren

Einige Parameter, die auf schreibgeschützte Technologieobjekt-Datenbausteinvariablen abgebildet sind, sind in der PublicAPI schreibbar. Diese Parameter werden nach ParameterData in der exportierten XML-Datei exportiert.

Einige Parameter sind nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet. Diese Parameter werden nach ParameterData in der exportierten XML-Datei exportiert.

Der Parametername, die zulässigen Werte und die Standardwerte sind die gleichen wie die Werte für die entsprechenden Datenbausteinvariablen.

Das Attribut ParameterData enthält die folgenden Parameter für TO_MeasuringInput:

- Parameter.MeasuringInputType
- _ListenToMeasuringInput

Der Parameter "_AssociatedObject" ist nicht direkt auf eine Technologieobjekt-Datenbausteinvariable abgebildet. Er wird nicht nach ParameterData in der exportierten XML-Datei exportiert.

XML-Element "Connection"

Das XML-Element Connection beschreibt die Verschaltungen eines Technologieobjekts. Die Namen und Werte der unten aufgeführten Attribute entsprechen den Parameternamen und Werten der bestimmten MeasuringInputHardwareConnectionProvider.

Wert	Beschreibung
Interface	Einziger möglicher Wert: MeasuringInput
InputAddress	• Der Wert ist eine Bitadresse. • Er beschreibt eine Verbindung zu einem DeviceItem oder einem channel.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

Spezifische Parameter für die Technologieobjekte Nocken und Nockenspur

Verwendung

Die API unterstützt den Import und Export von Technologieobjekten. Im folgenden Abschnitt werden die spezifischen Parameter beschrieben.

TO_OutputCam und TO_CamTrack exportieren

Einige Parameter, die auf schreibgeschützte Technologieobjekt-Datenbausteinvariablen abgebildet sind, sind in der PublicAPI schreibbar. Diese Parameter werden nach ParameterData in der exportierten XML-Datei exportiert. Der Parametername, die zulässigen Werte und die Standardwerte sind die gleichen wie die Werte für die entsprechenden Datenbausteinvariablen.

Für TO_OutputCam enthält das Attribut ParameterData den Parameter Interface.LogicOperation.

TO_CamTrack hat keine Elemente im Attribut ParameterData.

Der Parameter "_AssociatedObject" ist nicht direkt auf eine Technologieobjekt-Datenbausteinvariable abgebildet. Er wird nicht nach ParameterData in der exportierten XML-Datei exportiert.

XML-Element "Connection"

Das XML-Element Connection beschreibt die Verschaltungen eines Technologieobjekts. Die Namen und Werte der unten aufgeführten Attribute entsprechen den Parameternamen und Werten der bestimmten OutputCamHardwareConnectionProvider.

Wert	Beschreibung	TO_OutputCam	TO_CamTrack
Interface	Einziger möglicher Wert: OutputCam	X	-
OutputAddress	- Der Wert ist eine Bitadresse. - Er beschreibt eine Verbindung zu einem channel.	X	X
OutputTag	- Der Wert ist der Name der verschalteten PlcTag. - Er beschreibt eine Verbindung zu einer analogen Variablen.	X	X

Spezifische Parameter für Technologieobjekte Kurvenscheibe

Verwendung

Die API unterstützt den Import und Export von Technologieobjekten. Im folgenden Abschnitt werden die spezifischen Parameter beschrieben.

TO_Cam oder TO_CAM_10k exportieren

Das Kurvenscheibenprofil eines Technologieobjekts Kurvenscheibe des Typs TO_Cam oder TO_CAM_10k wird in das Element ParameterData exportiert.

Die Reihenfolge der folgenden Beschreibung von Elementen stellt die Struktur von ParameterData dar.

```

<ParameterData>
  <Parameters xmlns="http://www.siemens.com/automation/Openness/SW/Parameters/v1">
    <ProfileData xmlns="http://www.siemens.com/automation/Openness/SW/Motion/Cam/v1">
      <GeneralConfiguration StandardContinuity="Acceleration"
        StandardOptimizationGoal="Velocity"
        InterpolationMode="CubicSpline"
        BoundaryConditions="FirstDerivative">
        <DesignLeadingRange Start="0" End="360" />
        <DesignFollowingRange Start="-1" End="1" />
      </GeneralConfiguration>
      <Elements>
        <Point X="0" Y="0" />
        <Line StartX="0" EndX="45" StartY="0" EndY="1" />
        <Line StartX="45" EndX="90" StartY="1" EndY="1" />
        <Point X="120" Y="0" />
      </Elements>
    </ProfileData>
  </Parameters>
</ParameterData>

```

XML-Struktur von ParameterData

XML-Element	Beschreibung
ProfileData	Das XML-Element ProfileData ist ein einzelnes untergeordnetes Element von Parametern innerhalb des EOM-Attributs ParameterData. Erweiterungselement auf oberster Ebene, enthält das XML-Element GeneralConfiguration, gefolgt vom XML-Element Elements.
GeneralConfiguration	Das XML-Element GeneralConfiguration beschreibt die allgemeine Konfiguration, die für das vollständige Kurvenscheibenprofil gültig ist. Zusätzlich werden die folgenden optionalen Attribute verwendet: - Attribut StandardContinuity Mögliche Werte sind Position, Velocity, Acceleration (Standardwert) und Jerk. - Attribut StandardOptimizationGoal Mögliche Werte sind None (Standardwert), Velocity, Acceleration, Jerk und DynamicMoment. - Attribut InterpolationMode Mögliche Werte sind Linear, CubicSpline (Standardwert) und BezierSpline. - Attribut BoundaryConditions Mögliche Werte sind NoConstraint (Standardwert) und FirstDerivative.
DesignLeadingRange	Das XML-Element beschreibt den Leitwertbereich der Kurvendefinition. Die folgenden beiden optionalen Attribute mit Datentyp xsd:float werden verwendet: - Attribut Start Der Standardwert ist 0. - Attribut End Der Standardwert ist 360. Der Wert von Start muss kleiner als der Wert von End sein.

XML-Element	Beschreibung
DesignFollowingRange	Das XML-Element beschreibt den folgenden Wertebereich der Kurvendefinition. - Attribut Start Der Standardwert ist -1. - Attribut End Der Standardwert ist 1.
Elements	Das XML-Element enthält die Liste der Elemente des Kurvenscheibenprofils. Mögliche XML-Elemente: - Point - Line - Polynomial - VDITransition - Sine - InverseSine - PointGroup Mit Ausnahme des XML-Elements Point verwenden alle XML-Elemente die erforderlichen XML-Attribute StartX und EndX des Typs xsd:float. Diese Attribute enthalten die X-Koordinate am Anfang bzw. am Ende des Segments.
Point	Das XML-Element beschreibt einen einzelnen Punkt. Die folgenden Attribute werden verwendet, alle haben den Datentyp xsd:float und den Standardwert 0: - Erforderliches Attribut X - Erforderliches Attribut Y - Optionales Attribut Velocity - Optionales Attribut Acceleration - Optionales Attribut Jerk
Line	Das XML-Element beschreibt ein Line-Segment. Die folgenden Attribute werden verwendet, alle haben den Datentyp xsd:float und den Standardwert 0: - Optionales Attribut StartY Enthält die Y-Koordinate am Anfang der Zeile. - Optionales Attribut EndY Enthält die Y-Koordinate am Ende der Zeile. - Optionales Attribut Gradient Enthält den Gradienten der Zeile. Zwei dieser drei optionalen Attribute müssen in der export.XML vorhanden sein.
Polynomial	Das XML-Element beschreibt ein Polynomial-Segment. Das XML-Element Polynomial hat die Unterelemente TrigonometricValues, Coefficients und Constraints. TrigonometricValues kann optional verwendet werden, doch entweder Coefficients oder Constraints müssen vorhanden sein.

XML-Element	Beschreibung
TrigonometricValues	Das XML-Element TrigonometricValues kann optional als Unterelement von Polynomial verwendet werden. Es hat die folgenden Attribute mit dem Datentyp xsd:float: • Optionales Attribut Amplitude Der Standardwert ist 1. • Optionales Attribut StartPhase Der Standardwert ist 0. • Optionales Attribut EndPhase Der Standardwert ist 6,2831853071795862. • Optionales Attribut Frequency Der Standardwert ist 1. • Optionales Attribut PeriodLength Der Standardwert ist 1. Zwei der Attribute StartPhase, EndPhase, Frequency und PeriodLength müssen verwendet werden. Zusätzlich muss mindestens StartPhase oder EndPhase vorhanden sein.
Coefficients	Das XML-Element Coefficients kann optional als Unterelement von Polynomial verwendet werden. Es hat die optionalen Attribute C0, C1, C2, C3, C4, C5 and C6. Jedes davon hat den Datentyp xsd:float und den Standardwert 0.
Constraints	Das XML-Element kann optional als Unterelement von Polynomial verwendet werden. Es hat die folgenden Attribute mit dem Datentyp xsd:float und dem Standardwert 0: Optionales Attribut LeftValue Optionales Attribut RightValue Optionales Attribut LeftVelocity Optionales Attribut RightVelocity Optionales Attribut LeftAcceleration Optionales Attribut RightAcceleration Optionales Attribut LeftJerk Optionales Attribut RightJerk Optionales Attribut Lambda Zusätzlich hat es das optionale Attribut LambdaMode, das die Werte Relative und Absolute (Standardwert) unterstützt.
VDITransition	Das XML-Element beschreibt ein VDI-Transitionssegment. Es hat die optionalen Attribute LeftContinuity und RightContinuity, die die Werte AsProfile (Standardwert), Position, Velocity, Acceleration und Jerk haben können. Zusätzlich kann das optionale Attribut OptimizationGoal mit den möglichen Werten AsProfile (Standardwert), None, Velocity, Acceleration, Jerk und DynamicMoment verwendet werden.

XML-Element	Beschreibung
InverseSine	Das XML-Element beschreibt ein InverseSine-Segment. Es hat die folgenden optionalen Attribute: • InterpolationPointCount • MaxFollowingValueTolerance • MathStartX • MathEndX • Minimum • Maximum • Inversed
Sine	Das XML-Element beschreibt ein Sine-Segment. Das Sine-Segment hat die folgenden Attribute: • Amplitude • StartPhase • EndPhase • Frequency • PeriodLength • Inclination • StartOffset • EndOffset
PointGroup	Das XML-Element beschreibt ein PointGroup-Segment, das mehrere Punkte enthält. Das PointGroup-Element hat die folgenden optionalen Attribute: • ApproximationDataPoints • ApproximationTolerance • LeadingValueMode • FollowingValueMode • ApproximationMode

Spezifische Parameter für Technologieobjekte Kinematik

Verwendung

Die API unterstützt den Import und Export von Technologieobjekten. Im folgenden Abschnitt werden die spezifischen Parameter beschrieben.

TO_Kinematics exportieren

Einige Parameter, die auf schreibgeschützte Technologieobjekt-Datenbausteinvariablen abgebildet sind, sind in der PublicAPI schreibbar. Diese Parameter werden nach ParameterData in der exportierten XML-Datei exportiert. Der Parametername, die zulässigen Werte und die Standardwerte sind die gleichen wie die Werte für die entsprechenden Datenbausteinvariablen.

- Kinematics.TypeOfKinematics
- MotionQueue.MaxNumberOfCommands

Einige Parameter sind nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet. Dieser Parameter wird nach ParameterData in der exportierten XML-Datei exportiert. Der Parametername, die zulässigen Werte und die Standardwerte sind die gleichen wie die Werte für die entsprechenden Datenbausteinvariablen.

Die betroffenen Parameter sind in den folgenden Tabellen aufgelistet:

Name	Standardwert
_Units.LengthUnit	1013 (mm)
_Units.LengthVelocityUnit	1062 (mm/s)
_Units.AngleUnit	1126 (Nm)
_Units.AngleVelocityUnit	1120 (N)
_Properties.UseHighResolutionPositionValues ⁶⁾	false
_X_Minimum	0.0
_X_Maximum	0.0
_Y_Minimum	0.0
_Y_Maximum	0.0
_Z_Minimum	0.0
_Z_Maximum	0.0
_A3_Maximum	0.0

XML-Struktur von Connection

Das XML-Element Connection beschreibt die Verschaltungen eines Technologieobjekts.

Die Namen und Werte der unten aufgeführten Attribute entsprechen den Parameternamen und Werten der bestimmten AxisEncoderHardwareConnectionInterface.

XML-Element	Beschreibung
AdditionalData	Das Element enthält bis zu sechs KinematicsAxis-Elemente, gefolgt von einem ConveyorTrackingLeadingValues-Element.
KinematicsAxis	Das Element beschreibt eine verschaltete Kinematikachse. Dieses XML-Element hat die folgenden Attribute: • Erforderliches Attribut Index Die Werte des Index muss eindeutig sein. – Für < V7.0 lauten die möglichen Werte 1 ... 4 Die Indexwerte entsprechen den Kinematikachsen A1 bis A4. – Ab V7.0 lauten die möglichen Werte 1 ... 6 Die Indexwerte entsprechen den Kinematikachsen A1 bis A6. Die Kinematikachsen A5 und A6 sind nur mit "S7-1500T Motion Control KinPlus" verwendbar. • Erforderliches Attribut Ref Kennzeichnet den Namen des verschalteten Technologieobjekts Achse. Dieser Verweis muss eindeutig sein. • Erforderliches Attribut Type Enthält den Typ der verschalteten Achse. Die Version der verschalteten Achse ist nicht notwendig.
ConveyorTrackingLeadingValues	Das Element enthält die Elemente für SetPointCoupling, gefolgt von Elementen für ActualValueCoupling, DelayedCoupling.
SetPointCoupling	Das Element beschreibt ein verschaltetes Leitwert-TO, das über Sollwerte gekoppelt ist. Es hat die folgenden Attribute: • Erforderliches Attribut Ref Kennzeichnet den Namen des verschalteten Technologieobjekts, das den Leitwert für das Conveyor Tracking liefert. • Erforderliches Attribut Type Enthält den zugehörigen TO-Typ (versionsunabhängig).
ActualValueCoupling	Das Element beschreibt ein verschaltetes Leitwert-TO, das über Istwerte gekoppelt ist. Es werden die gleichen Attribute wie bei SetPointCoupling-Elementen verwendet.
DelayedCoupling	Das Element beschreibt ein verschaltetes Leitwert-TO, das über Istwerte gekoppelt ist. Es werden die gleichen Attribute wie bei SetPointCoupling-Elementen verwendet.

Spezifische Parameter für Technologieobjekte Interpreter

Verwendung

Die API unterstützt den Import und Export von Technologieobjekten. Im folgenden Abschnitt werden die spezifischen Parameter beschrieben.

TO_Interpreter exportieren

Der Parameter `Parameter.MaxNumberOfCommands` ist dem schreibgeschützten Technologieobjekt `Datenbausteinvariable` zugeordnet, das in der PublicAPI beschreibbar ist.

Der Parameter `Parameter.MaxNumberOfCommands` wird nach `ParameterData` in der exportierten XML-Datei exportiert. Der Parametername, die zulässigen Werte und die Standardwerte sind die gleichen wie die Werte für die entsprechenden Datenbausteinvariablen.

Spezifische Parameter für Technologieobjekte Interpreterprogramm

Verwendung

Die API unterstützt den Import und Export von Technologieobjekten.

Das Technologieobjekt `Interpreterprogramm` hat keine Parameter.

Programm TO_Interpreter exportieren

Der Quellcode des Technologieobjekts `Interpreterprogramms` wird in das XML-Element `SourceData` exportiert.

Die folgende Reihenfolge von Elementen stellt die Struktur von `ParameterData` dar.

```
<ParameterData>
  <Parameters xmlns="http://www.siemens.com/automation/Openness/SW/Parameters/v1">
    <SourceData xmlns="http://www.siemens.com/automation/Openness/SW/Motion/InterpreterProgram/v1">
      PROGRAM main
        VAR
          SetPointInWCS: ARRAY[1..6] OF TO_Struct_Ipr_Frame;
        END_VAR
        powerOn( axis := Axis_001 );
        powerOn( axis := Axis_002 );
        home( Axis_001, mode := 7, sensor := 0 );
        home( Axis_002, mode := 7, sensor := 0 );
        posAbs( Axis_001, 180.0, dir := 3, v := 100.0 );
        ...
      END_PROGRAM
    </SourceData>
  </Parameters>
</ParameterData>
```

6.4.3.7 PID-Regelung

Spezifische Attribute für PID_Compact

Die API-Schnittstelle unterstützt den Export und Import von Technologieobjekten. Eine Liste mit allen verfügbaren Variablen finden Sie im Funktionshandbuch "SIMATIC S7-1200/S7-1500 PID control" im Internet (<https://support.industry.siemens.com/cs/ww/de/view/108210036>).

Exportieren

Beim Technologieobjekt PID_Compact sind alle Parameter direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet. Alle Parameter haben den Datentyp und den Standardwert wie im Datenbaustein definiert. Das XML-Element `ParameterData` ist leer.

Import

Ab Version PID_Compact V3.0 zählen alle Parameter zu Bestandteilen des Imports.

Bis PID_Compact V2.4 sind die folgenden Parameter für PID_Compact nicht Bestandteil des Imports:

- PhysicalUnit
- PhysicalQuantity
- _Config.OutputSelect
- _Retain.CtrlParams.SetByUser

Diese Parameter haben nach dem Import den Standardwert.

Hinweis

Bis PID_Compact V2.4 stellen Sie die zugehörigen Einstellungen nach dem Import einer Exportdatei manuell wieder her. Das kann mit Hilfe der Funktion Parameter schreiben (Seite 560) geschehen.

Spezifische Attribute für PID_3Step und PID_Temp

Die API-Schnittstelle unterstützt den Export und Import von Technologieobjekten. Eine Liste mit allen verfügbaren Variablen finden Sie im Funktionshandbuch "SIMATIC S7-1200/S7-1500 PID control" im Internet (<https://support.industry.siemens.com/cs/ww/de/view/108210036>).

Exportieren

Bei den Technologieobjekten PID_3Step und PID_Temp sind alle Parameter direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet. Alle Parameter haben den Datentyp und den Standardwert wie im Datenbaustein definiert. Das XML-Element `ParameterData` ist leer.

Import

Die folgenden Parameter für PID_3Step und PID_Temp sind nicht Bestandteil des Imports:

- PhysicalUnit
- PhysicalQuantity

Diese Parameter haben nach dem Import den Standardwert. Wenn Sie sie ändern möchten, müssen Sie die Funktion Parameter schreiben (Seite 560) verwenden.

Hinweis

Stellen Sie die zugehörigen Einstellungen nach einem Import einer PID_3Step- oder PID_Temp-Exportdatei manuell wieder her.

6.4.3.8 Zählen

Die API-Schnittstelle unterstützt den Export und Import von Technologieobjekten. Eine Liste sämtlicher verfügbarer Parameter finden Sie in der Produktinformation "Parameter von Technologieobjekten in TIA Portal Openness" im Internet (<https://support.industry.siemens.com/cs/ww/de/view/109744932>).

Bei den TO-Typen High_Speed_Counter und SSI_Absolute_Encoder entsprechen alle Parameter Mitgliedern des TO DB und haben den Datentyp und den Standardwert wie im TO DB definiert. Das EOM-Attribut `ParameterData` ist leer.

6.4.3.9 Easy Motion Control

Die API-Schnittstelle unterstützt den Export und Import von Technologieobjekten. Eine Liste sämtlicher verfügbarer Parameter finden Sie in der Produktinformation "Parameter von Technologieobjekten in TIA Portal Openness" im Internet (<https://support.industry.siemens.com/cs/ww/de/view/109744932>).

Beim TO-Typ AXIS_REF entsprechen alle Parameter Mitgliedern des TO DB und haben den Datentyp und den Standardwert wie im TO DB definiert. Das EOM-Attribut `ParameterData` ist leer.

6.4.4 Variablentabellen

6.4.4.1 PLC-Variablentabellen exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Pro PLC-Variablentabelle wird eine XML-Datei exportiert.

Die Schnittstelle TIA Portal Openness API unterstützt den Export aller PLC-Variablentabellen aus der Systemgruppe und deren Untergruppen.

Programmcode

Ändern Sie den folgenden Programmcode, um alle PLC-Variablentabellen aus der Systemgruppe und deren Untergruppen zu exportieren:

```
private static void ExportAllTagTables(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    // Export all tables in the system group
    ExportTagTables(plcTagTableSystemGroup.TagTables);
    // Export the tables in underlying user groups
    foreach(PlcTagTableUserGroup userGroup in plcTagTableSystemGroup.Groups)
    {
        ExportUserGroupDeep(userGroup);
    }
}

private static void ExportTagTables(PlcTagTableComposition tagTables)
{
    foreach(PlcTagTable table in tagTables)
    {
        table.Export(new FileInfo(string.Format(@"D:\Samples\{0}.xml", table.Name)),
            ExportOptions.WithDefaults);
    }
}

private static void ExportUserGroupDeep(PlcTagTableUserGroup group)
{
    ExportTagTables(group.TagTables);
    foreach(PlcTagTableUserGroup userGroup in group.Groups)
    {
        ExportUserGroupDeep(userGroup);
    }
}
```

Siehe auch

Export von Projektierungsdaten (Seite 1253)

6.4.4.2 PLC-Variablentabelle importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Programmcode

Ändern Sie den folgenden Programmcode, um PLC-Variablentabellen oder eine Ordnerstruktur mit PLC-Variablentabellen aus einer XML-Datei in die Systemgruppe oder eine benutzerdefinierte Gruppe zu importieren:

```
//Imports tag tables to the tag system group
private static void ImportTagTable(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    PlcTagTableComposition tagTables = plcTagTableSystemGroup.TagTables;
    tagTables.Import(new FileInfo(@"D:\Samples\myTagTable.xml"), ImportOptions.Override);
    // Or, to import into a subfolder:
    // plcTagTableSystemGroup.Groups.Find("SubGroup").TagTables.Import(new
    FileInfo(@"D:\Samples\myTagTable.xml"), ImportOptions.Override);
}
```

Siehe auch

Anmerkungen zur Leistung von TIA Portal Openness (Seite 131)

6.4.4.3 Einzelne Variable oder Konstante aus einer PLC-Variablentabelle exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Die API-Schnittstelle unterstützt den Export einer Variablen oder Konstanten aus einer PLC-Variablentabelle in eine XML-Datei. Achten Sie dabei darauf, dass die verwendeten Variablentabellennamen den Dateibenennungskonventionen Ihres Dateisystems entsprechen.

Sie können einen Export mit der Exporteinstellung `ExportOptions.None` vornehmen. Bei dieser Einstellung wird Folgendes nicht exportiert:

- Variable: `ExternalAccessible` (Standardwert: true)
- Variable: `ExternableWritable` (Standardwert: true)
- Variable: `ExternalVisible` (Standardwert: true)
- Variable: `LocalAddress` (Standardwert: leer)
- Konstante: `Value` (Standardwert: leer)
- Variable und Konstante: `DataTypeName` (Standardwert: leerer String)
- Variable: `IsSafety` (true, wenn sicherheitsrelevant; andernfalls false)

Der Kommentar einer Variablen oder Konstanten wird nur exportiert, wenn mindestens eine Sprache für den Kommentar festgelegt ist. Wenn der Kommentar nicht für alle Projektsprachen festgelegt ist, wird dieser Kommentar nur für die festgelegten Projektsprachen exportiert.

Hinweis**PLC-Systemkonstanten**

PLC-Systemkonstanten werden vom Export und Import ausgeschlossen.

Programmcode

Um eine bestimmte Variable oder Konstante aus einer PLC-Variablentabelle in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a single tag or constant of a controller tag table
private static void ExportTag(PlcSoftware plcSoftware, string tagName)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    PlcTag tag = plcTagTableSystemGroup.TagTables[0].Tags.Find(tagName);
    if(tag != null)
    {
        tag.Export(new FileInfo(string.Format(@"D:\Samples\{0}.xml", tag.Name)),
            ExportOptions.WithDefaults);
    }
}

private static void ExportUserConstant(PlcSoftware plcSoftware, string userConstantName)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    PlcUserConstant plcConstant =
        plcTagTableSystemGroup.TagTables[0].UserConstants.Find(userConstantName);
    if(plcConstant != null)
    {
        plcConstant.Export(new FileInfo(string.Format(@"D:\Samples\{0}.xml", plcConstant.Name)),
            ExportOptions.WithDefaults);
    }
}
```

Siehe auch

Export von Projektierungsdaten (Seite 1253)

Anmerkungen zur Leistung von TIA Portal Openness (Seite 131)

6.4.4.4 Einzelne Variable oder Konstante in eine PLC-Variablentabelle importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Verwendung

Sie können in einem Importaufruf entweder Variablen oder Konstanten importieren.

Hinweis

Konstanten können nur als Anwenderkonstanten importiert werden.

Programmcode

Ändern Sie den folgenden Programmcode, um Variablengruppen oder einzelne Variablen und Konstanten aus einer XML-Datei zu importieren:

```
//Imports tags into a plc tag table
private static void ImportTag(PlcSoftware plcSoftware, string tagtableName)
{
    PlcTagTableSystemGroup plcTagTableSystemgroup = plcSoftware.TagTableGroup;
    PlcTagTable tagTable = plcTagTableSystemgroup.TagTables.Find(tagtableName);
    if(tagTable == null) return;

    tagTable.Tags.Import(new FileInfo(@"D:\Samples\myTags.xml"), ImportOptions.Override);
}

//Imports constants into a plc tag table
private static void ImportConstant(PlcSoftware plcSoftware, string tagtableName)
{
    PlcTagTableSystemGroup plcTagTableSystemgroup = plcSoftware.TagTableGroup;
    PlcTagTable tagTable = plcTagTableSystemgroup.TagTables.Find(tagtableName);
    if(tagTable == null) return;

    tagTable.UserConstants.Import(new FileInfo(@"D:\Samples\myConstants.xml"),
    ImportOptions.Override);
}
```

Siehe auch

Export von Projektierungsdaten (Seite 1253)

Anmerkungen zur Leistung von TIA Portal Openness (Seite 131)

6.5 Hardware-Daten importieren/exportieren

6.5.1 AML-Dateiformat

Einleitung

AutomationML ist ein neutrales, auf XML basierendes Datenformat zum Speichern und Austauschen von Engineering-Informationen einer Anlage und wird als offener Standard angeboten. Ziel von AutomationML ist die Verschaltung der heterogenen Werkzeuglandschaft moderner Engineering-Tools in deren unterschiedlichen Disziplinen wie mechanisches Anlagen-Engineering, Auslegung der Elektrik, HMI, PLC, Robotersteuerung.

Das für den Export und Import von CAx-Daten verwendete Klassifizierungsmodell basiert auf den folgenden AML-Standards:

- Whitepaper AutomationML Teil 1 – AutomationML Architecture, November 2018
- Whitepaper AutomationML Teil 2 – AutomationML Role Class Libraries, Oktober 2014
- Whitepaper AutomationML – AutomationML Communication, September 2014
- Whitepaper AutomationML – AutomationML and eCl@ss Integration, November 2021
- Best Practice Recommendation Multilingual expressions in AutomationML, März 2017
- Best Practice Recommendation Modelling of Reference Designations, September 2017

Schema

Das AutomationML-Datenaustauschmodell wird durch das CAEX-Schema Version 2.15 beschrieben.

6.5.2 Pruned AML

Einleitung

Pruning beschreibt den Vorgang der Optimierung des Inhalts durch Entfernen bestimmter Dinge, die nicht unbedingt angegeben werden müssen. Bei externen Werkzeugen wie EPLAN haben die automatisch erstellten Submodulinformationen in einer Hardware-Konfiguration keine Bedeutung hinsichtlich EPLAN. Deshalb generieren diese Werkzeuge eine AML-Datei, indem die automatisch erstellten Submodulinformationen aus der Hardware-Konfiguration entfernt werden. Diese Datei wird Pruned AML genannt.

Pruned AML generieren

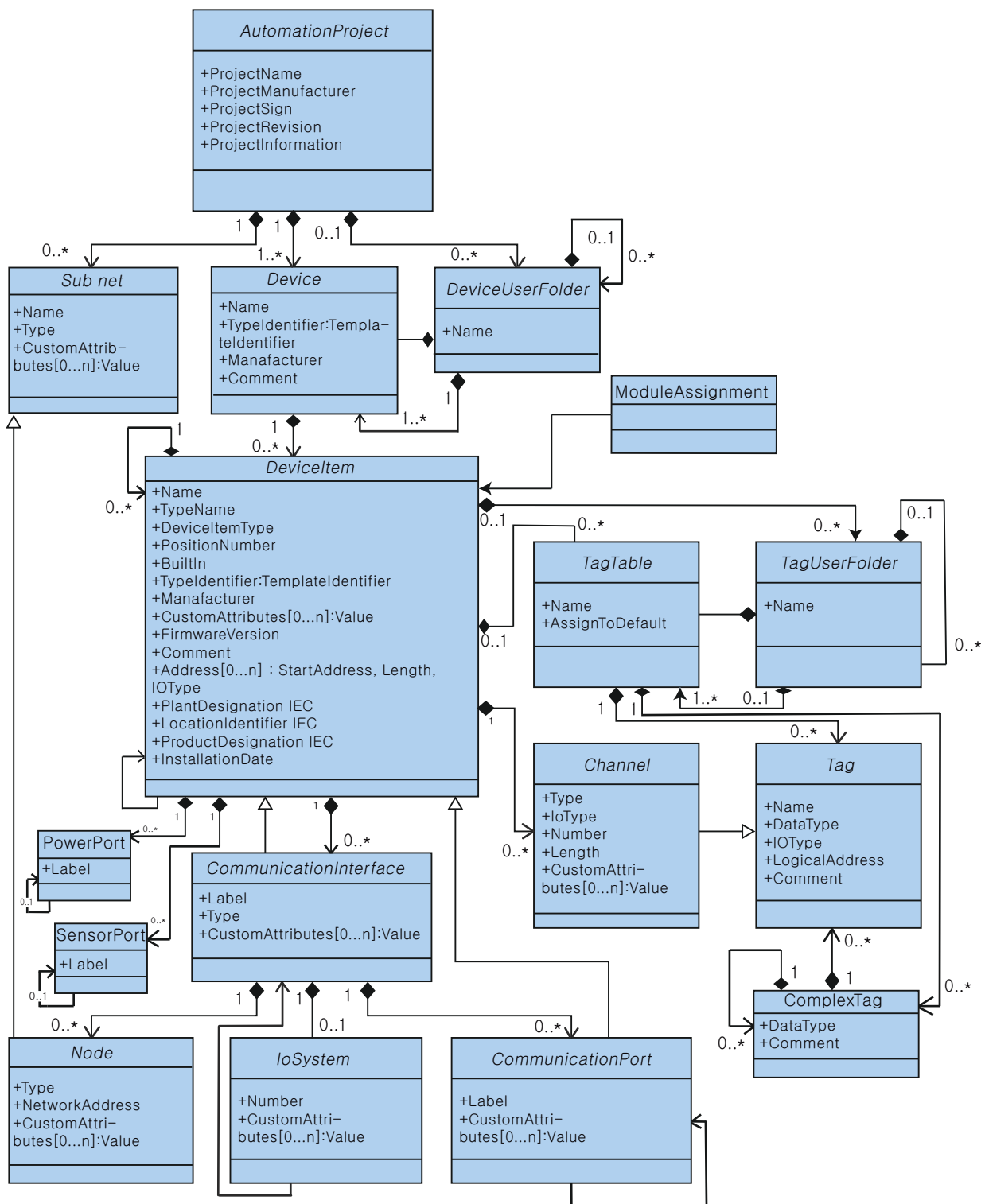
Die Generierung von Pruned AML basiert auf den folgenden Regeln in dieser Reihenfolge.

1. Wenn ein Geräteelement steckbar ist, wird kein Pruning durchgeführt.
2. Wenn ein Geräteelement vom Typ "Schnittstelle" oder "Port" ist, wird kein Pruning durchgeführt.
3. Wenn ein Geräteelement integriert ist und sich auf Baugruppenträgerebene befindet, wird kein Pruning durchgeführt.
4. Adressobjekte vom Typ "Diagnose" sind für den Pruning-Algorithmus nicht relevant.
5. Adressobjekte, die mit den automatisch erstellten Submodulen verknüpft sind, werden unter dem direkt übergeordneten Element angegeben (bei dem es sich um ein nicht automatisch erstelltes Submodul handeln muss).
6. Adressobjekte werden in die gleiche Sequenz eingeschlossen, die auch von TIA Portal Openness ausgegeben wird.

6.5.3 Übersicht über Objekte und Parameter von CAx-Import/-Export

Export/Import-Objekte und -Attribute

Die folgende Abbildung zeigt exportierbare Objekte mit ihren Attributen und Abhängigkeiten im CAx-Import/Export.



6.5.4 Struktur der CAx-Daten zum Import/Export

Grundstruktur einer Exportdatei

Die Exportdatei wird in einem AML-Format erzeugt. Die AML-Datei beginnt mit einer Dokumentinformation.

Die Exportdatei umfasst die folgenden zwei Abschnitte:

- Weitere Informationen
Der Abschnitt enthält Informationen über das Schreibtool, die referenzierten Dokumentversionen (Dokumente, die den Inhalt der AML-Datei definieren) usw.

Die nachfolgende XML-Datei zeigt die AML-Datei mit der neuesten AR APC-Empfehlung entsprechend den aus TIA Portal V19 exportierten Dateiinformationen, die den Abschnitt <AdditionalInformation> zeigen.

```
<?xml version="1.0" encoding="utf-8"?>
<CAEXFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" FileName="Project82.aml"
SchemaVersion="2.15" xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
<AdditionalInformation>
<WriterHeader>
<WriterName>Totally Integrated Automation Portal</WriterName>
<WriterID>1d4fceb-1ad6-4881-b01d-bca335d94a46:V1.0</WriterID>
<WriterVendor>Siemens AG</WriterVendor>
<WriterVendorURL>www.siemens.com</WriterVendorURL>
<WriterVersion>19</WriterVersion>
<WriterRelease>1900.0000.0.0</WriterRelease>
<LastWritingDateTime>2023-07-19T04:21:32.9174411Z</LastWritingDateTime>
</WriterHeader>
</AdditionalInformation>
<AdditionalInformation AutomationMLVersion="2.0" />
<AdditionalInformation DocumentVersions="Recommendations">
<Document DocumentIdentifier="AR APC" Version="1.4.0" />
</AdditionalInformation>
...
...
</CAEXFile>
```

Hinweis

CAx exportiert und importiert die Version von AR APC in eine AML-Datei gemäß der installierten Version des TIA Portals.

- Instanzhierarchie
Dieser Abschnitt enthält die hierarchische Reihenfolge der exportierten internen Elemente.

```
...
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
<InternalElement ID="d4dc896a-f4a5-41b6-9c48-8d3a0a5a4343" Name="CAx_asterisk_AML_03_V14">
  <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
  <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
  <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
  <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
  <InternalElement ID="544f3a69-5f65-45ba-ac2f-1448db9493fd" Name="PN/IE_1">
    ...
  </InternalElement>

  <InternalElement ID="12116ac0-94b7-49d2-888d-7d39bbc0caf5" Name="S71500/ET200MP station_1">
    ...
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
    "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
  <InternalLink Name="Link To Port_1" RefPartnerSideA=
    "726148ce-de5b-4728-8886-4ba273435479:CommunicationPortInterface" RefPartnerSideB=
    "cb9d24f3-8200-4c89-9b40-24ae850e293e:CommunicationPortInterface" />
  <InternalLink Name="Link To Port_2" RefPartnerSideA=
    "cb9d24f3-8200-4c89-9b40-24ae850e293e:CommunicationPortInterface" RefPartnerSideB=
    "726148ce-de5b-4728-8886-4ba273435479:CommunicationPortInterface" />
  ....
  <InternalLink Name="Link To IoSystem_3" RefPartnerSideA=
    "d8f6e006-3778-4a05-aab1-df844fe822fe:LogicalEndPoint_Interface" RefPartnerSideB=
    "2344b7af-329c-4215-92d1-6143b4627b56:LogicalEndPoint_IoSystem" />
</InternalElement>
</InstanceHierarchy>
</CAEXFile>
```

Interne Elemente

Alle Objekte innerhalb der Instanzhierarchie der AML-Datei sind `InternalElements`. Das interne Element `AutomationProject` enthält alle internen Elemente aller Rollenklassifizierungen. Jedes interne Element unterstützt ein Set von Attributen.

Das Attribut `<TypeIdentifier>` identifiziert jeden Objekttyp eines Hardware-Objekts, das über TIA Portal Openness erstellt werden kann.

Hinweis

Automatisch erzeugte Objekte

Automatisch erzeugte Objekte können nur von anderen Objekten erzeugt werden. Sie haben keine Attribute und keine Typkennung. Sie werden in die exportierte Datei aufgenommen, doch Sie können den Export eines bestimmten automatisch erstellten Objekts nicht anstoßen.

Am Ende des AML-Elements eines internen Elements wird Folgendes definiert:

- Rollenklassifizierung

Das Element `SupportedRoleClass` definiert den Objekttyp eines internen Elements. Der Objekttyp wird in der Rollenklassifizierungsbibliothek definiert, die den Standard-AML zu dem Objektmodell von TIA Portal Openness und TIA Portal abbildet.

```
...
<InternalElement ID="1d1a37ed-19d9-4a23-bc91-51f5a8e0244b" Name="Ungrouped devices">
  <InternalElement ID="ab193f5d-0375-4a6d-a576-a903e2b77cca" Name="ET 200SP station_1">
    ...
    <InternalElement ID="72d41729-90a7-4de3-9708-a8eeda6b1886" Name="IO device_1">
      ...
      <SupportedRoleClass RefRoleClassPath="
        AutomationProjectConfigurationRoleClassLib/DeviceItem" />
    </InternalElement>
    ...
    <SupportedRoleClass RefRoleClassPath="
      AutomationProjectConfigurationRoleClassLib/Device" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath="
    AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
</InternalElement>
...
```

- Interne Links

Das Element `InternalLink` definiert die Kommunikationspartner einer Verbindung.

```
...
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
  <InternalElement ID="d4dc896a-f4a5-41b6-9c48-8d3a0a5a4343" Name="CAx_asterisk_AML_03_V14">
    <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
    <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
    <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
    <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
    ...
    <SupportedRoleClass RefRoleClassPath="
      AutomationProjectConfigurationRoleClassLib/AutomationProject" />
    <InternalLink Name="Link To Port_1" RefPartnerSideA="
      726148ce-de5b-4728-8886-4ba273435479:CommunicationPortInterface" RefPartnerSideB="
      cb9d24f3-8200-4c89-9b40-24ae850e293e:CommunicationPortInterface" />
    <InternalLink Name="Link To Port_2" RefPartnerSideA="
      cb9d24f3-8200-4c89-9b40-24ae850e293e:CommunicationPortInterface" RefPartnerSideB="
      726148ce-de5b-4728-8886-4ba273435479:CommunicationPortInterface" />
    <InternalLink Name="Link To Port_3" RefPartnerSideA="
      65307e3e-95fd-41ac-9982-e5e4ffc2fb15:CommunicationPortInterface" RefPartnerSideB="
      58b1a3f2-f94b-48d1-ab5e-fbc4857cdfbc:CommunicationPortInterface" />
    <InternalLink Name="Link To Port_4" RefPartnerSideA="
      58b1a3f2-f94b-48d1-ab5e-fbc4857cdfbc:CommunicationPortInterface" RefPartnerSideB="
      65307e3e-95fd-41ac-9982-e5e4ffc2fb15:CommunicationPortInterface" />
    ...
  </InternalElement>
</InstanceHierarchy>
</CAEXFile>
```


Attribute

Attribute werden internen Elementen wie folgt zugeordnet:

```
...
<InternalElement ID="1d1a37ed-19d9-4a23-bc91-51f5a8e0244b" Name="Ungrouped devices">
  <InternalElement ID="ab193f5d-0375-4a6d-a576-a903e2b77cca" Name="ET 200SP station_1">
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>System:Device.ET200SP</Value>
    </Attribute>
    <InternalElement ID="7636c362-a7af-47bb-8a18-e6428a6d61ff" Name="Rack_0">
      <Attribute Name="TypeName" AttributeDataType="xs:string">
        <Value>Rack</Value>
      </Attribute>
      <Attribute Name="PositionNumber" AttributeDataType="xs:int">
        <Value>0</Value>
      </Attribute>
      <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
        <Value>False</Value>
      </Attribute>
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>System:Rack.ET200SP</Value>
      </Attribute>
      ...
      <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/Device" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
    "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
  <InternalLink Name="Link To Port_1" RefPartnerSideA=
    "5758e2ff-3974-41e9-8bcc-b61a23f1bb58:CommunicationPortInterface"
    RefPartnerSideB="46683602-5129-4504-a9d1-48e6421e2cf0:CommunicationPortInterface" />
  ...
</InternalElement>
```

Handhabung von Attributen

Die Handhabung von Attributen wird für jedes Attribut einzeln wie folgt definiert:

- Ignoriert
Das Attribut wird beim Import ignoriert und ist in der Exportdatei nicht vorhanden.
- Obligatorisch
Das Attribut muss in einer Importdatei vorhanden sein und kann in der Exportdatei nicht gelöscht werden.
- Optional
Falls dieses Attribut in der Importdatei fehlt, wird der Standardwert für das Attribut festgelegt. Dieses Attribut fehlt in der Exportdatei, wenn es für ein Objekt nicht verwendbar ist, z. B. haben nicht alle Module eine Firmware-Version.

- Nur Exportieren
Das Attribut wird intern durch das TIA Portal bestimmt, z. B. der Typenname des Geräteelements. Wenn dieses in einer Importdatei vorhanden ist, wird es durch das TIA Portal während des Imports ignoriert.
- Nur Importieren
Das Attribut kann das Importverhalten beeinflussen. Falls dieses Attribut in der Importdatei fehlt, wird das Verhalten dem Standardwert für das Attribut entsprechen.

Siehe auch

AML-Typkennungen (Seite 1486)

6.5.5 AML-Typkennungen

Interne Elemente

Der String `TypeIdentifizier` besteht aus den folgenden Teilen:

- `<TypeIdentifizierType>:<Identifizier>`

Die folgenden Werte für `TypeIdentifizierType` sind hier zulässig:

- `OrderNumber` dient der Angabe von physischen Modulen
- `GSD` dient der Angabe von GSD/GSDML-basierten Geräten
- `System` dient der Angabe von Systemen und generischen Geräten

Typkennung: OrderNumber

`OrderNumber` ist die allgemeine Typkennung für alle Module im Hardware-Katalog, ausgenommen GSD. AML-Typkennungen entsprechen nicht immer den TIA Portal Openness-Typkennungen. AML-Typkennungen haben keine Information zur `FirmwareVersion`. Die Information zur Firmwareversion ist im getrennten AML-Attribut "`FirmwareVersion`" enthalten.

Das Format für diese Typkennung ist eines der folgenden:

- `<OrderNumber>`
Beispiel: Bestellnummer: 3RK1 200-0CE00-0AA2

Hinweis

Platzhalter in den Bestellnummern

Es gibt einige wenige Module im Hardware-Katalog, die „Platzhalter“-Zeichen in ihren Bestellnummern verwenden, die eine bestimmte Gruppe realer Hardware repräsentieren, z. B. die verschiedenen Längen der S7-300-Baugruppenträger.

In diesem Fall können die spezifischen `OrderNumber` und die „Platzhalter“ `OrderNumber` beide verwendet werden, um eine Instanz des Hardware-Objekts zu erstellen. Sie können die Platzhalter jedoch nicht beliebig an jeder Stelle verwenden. Beispiel: Ein S7-300-Baugruppenträger kann auf die folgenden Arten erstellt werden:

Bestellnummer: 6ES7 390-1***0-0AA0

oder

Bestellnummer: 6ES7 390-1AE80-0AA0

Beachten Sie, dass Sie die folgende Struktur beispielsweise nicht verwenden dürfen:

Bestellnummer: 6ES7 390-1AE80-0A*0

Der Rückgabewert der ausgelesenen Typkennung ist immer die Bestellnummer aus dem Hardwarekatalog.

Beispiel: Bestellnummer: 6ES7 390-1AE80-0AA0 entspricht Bestellnummer: 6ES7 390-1***0-0AA0

Typkennung: GSD

Die Typkennung für GSD- und GSDML-basierte Geräte lautet `TypeIdentifier = GSD:<Identifizier>`

Die Kennung besteht aus folgenden Elementen

- `GsdName`: Name der GSD oder GSDML in Großbuchstaben
- `GsdType`: Ist eines der folgenden:
 - D: Gerät (Device)
 - R: Baugruppenträger (Rack)
 - DAP: Kopfmodul
 - M: Modul
 - SM: Submodul
- `GsdId`: ID der GSD oder GSDML

Die folgenden Formate für die Typkennungen werden beim CAx-Import/-Export unterstützt:

- `GSD.<GsdName>/<GsdType>`
Beispiele:
`GSD:SIEM8139.GSD/DAP`
`GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCP-20140313.XML/D`
- `<GsdName>/<GsdType>/<GsdId>`
Beispiel:
`GSD:SIEM8139.GSD/M/4`
`GSD:GSDML-V2.31-SIEMENS-SINAMICS_G110M-20140704.XML/M/IDM_DRIVE_47`

Typkennung: System

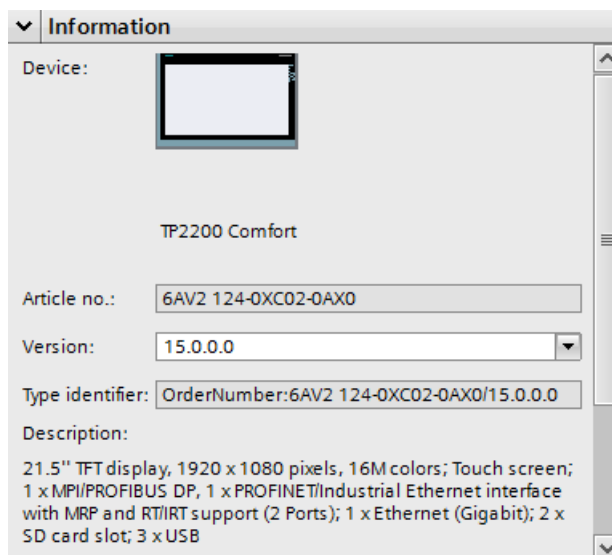
`System.` ist die Kennung für Objekte, die nicht durch andere Kennungen bestimmt werden können. Die Formate für diese `TypeIdentifizierType` sind die folgenden:

- `<SystemTypeIdentifizier>`
Beispiele:
`System:Device.S7300`
`System:Subnet.Ethernet`
- `<SystemTypeIdentifizier>/<AdditionalTypeIdentifizier>`
Die `AdditionalTypeIdentifizier` ist erforderlich, wenn `SystemTypeIdentifizier` nicht eindeutig ist.
Die `SystemTypeIdentifizier` hat ein Präfix für bestimmte Objekttypen:
`Subnet.`
`Device.`
`Rack.`
Beispiel: `System:Rack.S71600/Large`
Ein Baugruppenträger mit Bestellnummer wird über die `OrderNumber` identifiziert.

Typkennungen im TIA Portal anzeigen

Wenn Sie eine Typkennung kennen müssen, ermitteln Sie sie im TIA Portal wie folgt:

1. Aktivieren Sie unter "Optionen > Einstellungen > Hardware-Konfiguration > Anzeige der Typkennung" die Einstellung "Anzeige der Typkennung für Geräte und Module".
2. Öffnen Sie den Editor "Geräte & Netze".
3. Wählen Sie ein Gerät im Katalog aus.
Die Typkennung wird unter "Information" angezeigt.



6.5.6 Exportieren/Importieren von Informationen zur Basiseinheit über AML

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist offline.

Verwendung

Im TIA Portal können Sie Informationen zur Basiseinheit aus dem TIA Portal exportieren und in das TIA Portal importieren, um den Informationsaustausch mit anderen Tools wie EPLAN zu vereinfachen.

Beim Export aus ein und Import in ein TIA Portal-Projekt werden zwei Arten von Basiseinheiten unterstützt:

- Einzelne Basiseinheit
- Doppelte Basiseinheit

Export einer Basiseinheit

Beispiel: CAx exportiert für ein Modul, das mit Informationen zur Basiseinheit in TIA Portal konfiguriert ist, die Informationen zur Basiseinheit als "Submodul" unter einem Modul in der AML-Datei.

Das Submodul der Basiseinheit wird stets exportiert mit:

- PositionNumber: 0
- DeviceItemType: Zubehör
- BuiltIn: False
- TypeIdentifier: "Typ-ID der Basiseinheit"
- ID: Immer ein zufällig generierter GUID

Export einer einzelnen Basiseinheit

Das untenstehende Beispiel zeigt eine AML-Datei für ein DI-Modul, das mit einer einzelnen Basiseinheit im TIA Portal konfiguriert ist.

```
<InternalElement ID="6f76c890-5c5d-41c4-9ade-96543b0222ac" Name="DI 8x24VDC ST_1">
...
<InternalElement ID="69233c1f-7ef7-4999-8e84-691d0ff3a210" Name="BaseUnit">
<Attribute Name="DeviceItemType" AttributeDataType="xs:string">
<Value>Accessory</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 193-6BP00-0DA0</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
...
```

Export einer doppelten Basiseinheit

Das untenstehende Beispiel zeigt eine AML-Datei für zwei DI-Module, die mit doppelter Basiseinheit im TIA Portal konfiguriert sind. Beispiel: Das erste Modul mit der Basiseinheit wird mit dem Präfix IX300 und das zweite mit derselben doppelten Basiseinheit wird mit dem Präfix IX301 konfiguriert.

Beim Export wird nur das erste Modul der Basiseinheit mit dem Präfix IX300 mit dem Submodul der Basiseinheit in die AML-Datei exportiert. Das zweite Modul, das mit dem Präfix IX301, wird in kein untergeordnetes Submodul eingefügt.

```
<InternalElement ID="6f76c890-5c5d-41c4-9ade-96543b0222ac" Name="DI 8x24VDC ST_1">
...
<InternalElement ID="3a1bee8a-12d0-4ec4-849c-333d45113d9c" Name="BaseUnit">
<Attribute Name="DeviceItemType" AttributeDataType="xs:string">
<Value>Accessory</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 193-6BP60-0DA0</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
...
<InternalElement ID="5f843491-b053-4dc8-b879-9ac327ee2a7e" Name="DI 8x24VDC ST_2">
...
<InternalElement ID="55c30280-6f8a-4c37-9b2d-41bb90941258" Name="DI 8x24VDC ST_2">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value> </Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>>true</Value>
</Attribute>
...

```

Import einer Basiseinheit

Es ist möglich, ein mit einem Submodul einer Basiseinheit konfigurierten Modul in eine AML-Datei zu importieren.

Während des Imports einer Basiseinheit

- haben GUID, PositionNumber, BuiltIn und DeviceItem keine Relevanz und werden deshalb nicht im TIA Portal angezeigt.
- Wenn ein unbeabsichtigtes Modul in der AML-Datei ein Submodul einer Basiseinheit hat, gibt Openness das Attribut "BaseUnit" dafür nicht zurück. Somit zeigt CAX in der Protokolldatei eine entsprechende Warnung an.
- CAX prüft nicht auf die Richtigkeit der MLFB der Basiseinheit in der AML-Datei (außer zur Ermittlung, ob es sich um eine einzelne oder doppelte Basiseinheit handelt). CAX versucht, die MLFB der Basiseinheit über Openness festzulegen. Tritt in Openness ein Fehler auf, wird ein entsprechender Fehler angezeigt.
- Der Import der Vorgängerversion der AML-Datei mit/ohne Informationen zur Basiseinheit ist erfolgreich, wenn Informationen ins TIA Portal importiert werden.

Import einer einzelnen Basiseinheit

Beim Import einer einzelnen Basiseinheit wird nur das Modul, das das Submodul der Basiseinheit in der AML-Datei enthält, mit Informationen zur Basiseinheit ins TIA Portal importiert.

Die einzelne Basiseinheit wird anhand des Typidentifizier und dem nachfolgenden Muster in der AML-Datei identifiziert:

OrderNumber:xxxx **193-6[B|U|T]xYx-xxxx**, dabei liegt der Wert von Y (11. Stelle) im Bereich von 0 bis 5.

Import einer doppelten Basiseinheit:

Beim Import einer doppelten Basiseinheit werden zwei (benachbarte) Module mit Informationen zur Basiseinheit im TIA Portal angezeigt. Das erste Modul mit einem Submodul einer doppelten Basiseinheit in der AML-Datei wird mit einer doppelten Basiseinheit importiert, indem das Suffix |X300 an die MLFB der Basiseinheit angehängt wird. Das zweite Modul, das kein untergeordnetes Submodul einer Basiseinheit hat, wird mit derselben Basiseinheit importiert, indem das Suffix |X301 an die MLFB der Basiseinheit angehängt wird. Die doppelte Basiseinheit wird anhand des Typidentifizier und dem nachfolgenden Muster in der AML-Datei identifiziert:

OrderNumber:xxxx **193-6[B|U|T]x6x-xxxx**

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.7 Export/Import einer AML-Datei mit Verbindung zum Erweiterungsbaugruppenträger

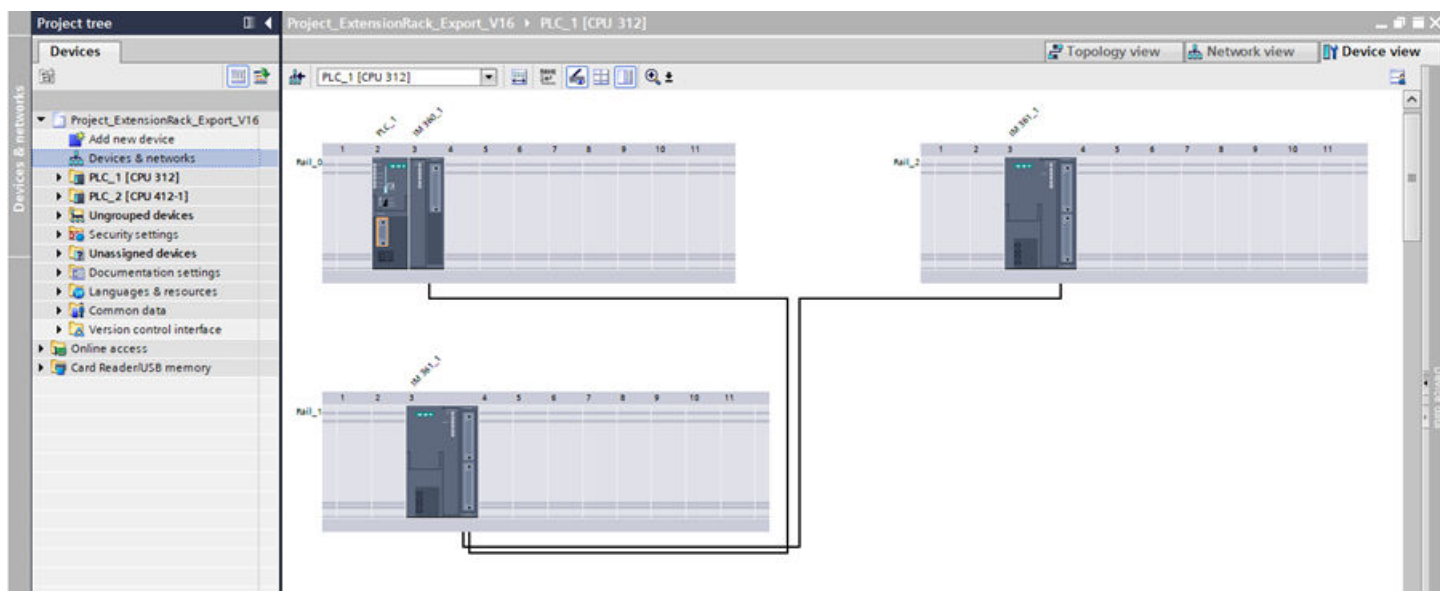
Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Im TIA Portal können Sie die Geräte mit mehreren Baugruppenträgern, die eine Erweiterungsbaugruppenträgerverbindung zu einer AML-Datei haben, exportieren und auch wieder importieren, um die gleiche Gerätekonfiguration zu erhalten, die im TIA Portal-Projekt angelegt wurde.

Beispiel: Gerät mit mehreren Baugruppenträgern mit Erweiterungsbaugruppenträgerverbindungen



AML-Struktur

Im TIA Portal werden Erweiterungsbaugruppenträgerverbindungen zwischen mehreren Baugruppenträgern direkt unter Sender- und Empfängermodulen modelliert (beide sind DeviceItem-Objekte). Entsprechend der AR APC-Empfehlung werden diese Verbindungen jedoch als Port-zu-Port-Verbindungen unter einem Kommunikationsport modelliert. Um diese Empfehlung zu erfüllen, wird unter dem IM-Modul ein Dummy-CommunicationInterface mit Dummy-CommunicationPort eingefügt.

Das folgende Beispiel zeigt einen Ausschnitt der Elementstruktur der exportierten AML-Datei für obige Gerätekonfiguration:

```

<InternalElement ID="1ddb8d5c-d6cc-42c9-b1d8-621219b139f6" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ExtensionRack</Value>
</Attribute>
...
<InternalElement ID="f25e531a-1793-4896-ade5-a87bd98de06e" Name="IM 46x SenderPort_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="9a824a06-89b9-4ba8-bee0-83c89b1f5e53"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
...
<InternalElement ID="d841278f-558a-41ab-9f03-91eeb454dc6b" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ExtensionRack</Value>
</Attribute>
...
<InternalElement ID="2e1ff3b2-8a3e-4d5b-a95c-3ed027287db9" Name="IM 46x ReceiverPort_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="98460c75-a05d-4c23-8f88-33878ccd79c5"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="7615a32e-09dc-4171-88ed-118026357bae" Name="IM 46x SenderPort_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X2</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>

```

6.5 Hardware-Daten importieren/exportieren

```

<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="846093a9-6473-4946-acf4-95a7813924df"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
...
<InternalElement ID="f4eb31c6-41d4-4a6e-ac33-de9c054a8c74" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ExtensionRack</Value>
</Attribute>
...
<InternalElement ID="c11d2227-91db-41ae-9d94-d822e3ab9c7a" Name="IM 46x ReceiverPort_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="a9b2cce1-7078-4347-b5f2-428dalad5326"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
...
<InternalLink Name="Link To Port_1" RefPartnerSideA="f25e531a-1793-4896-ade5-
a87bd98de06e:CommunicationPortInterface" RefPartnerSideB="2e1ff3b2-8a3e-4d5b-
a95c-3ed027287db9:CommunicationPortInterface" />
<InternalLink Name="Link To Port_2"
RefPartnerSideA="7615a32e-09dc-4171-88ed-118026357bae:CommunicationPortInterface"
RefPartnerSideB="c11d2227-91db-41ae-9d94-d822e3ab9c7a:CommunicationPortInterface" />

```

Hinweis

Die Erweiterungsbaugruppenträgerschnittstelle wird nur dann exportiert, wenn Erweiterungsbaugruppenträgerverbindungen vorhanden sind. Auch basiert die Anzahl der Dummy-Ports unterhalb der Dummy-Erweiterungsbaugruppenträgerschnittstelle auf den an der Baugruppenträgerverbindung auf Modulebene teilnehmenden Ports. Im obigen Beispiel unterstützt das Modul "IM 460-0_1" zwei Ports (IM 46x SenderPort_1 und IM 46x SenderPort_2). Im TIA Portal ist jedoch nur ein Port mit Verbindung konfiguriert. Deshalb enthält die exportierte AML-Datei nur einen Port unterhalb der Erweiterungsbaugruppenträgerschnittstelle.

Erweiterungsbaugruppenträgerverbindung

Die XML-Darstellung von Erweiterungsbaugruppenträgerverbindungen zwischen mehreren Baugruppenträgern geschieht im nachfolgend dargestellten Format.

ExternalInterface-

Das interne Element <ExternalInterface> muss unter dem internen Element <CommunicationPort> hinzugefügt werden, das an der Verbindung teilnimmt.

```
<InternalElement ID="[IM Module Unique ID]" Name="[IM Module Name]">
...
<InternalElement ID="[Dummy Interface Unique ID]" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ExtensionRack</Value>
</Attribute>
...
<InternalElement ID="[Dummy Port Unique ID]" Name="[IM Module Sender/Receiver Name]">
...
<ExternalInterface ID="[External Interface Unique ID]" Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="[Dummy Port Unique ID]" Name="[IM Module Sender/Receiver Name]">
...
<ExternalInterface ID="[External Interface Unique ID]" Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
```

Interner Link

Erweiterungsbaugruppenträgerverbindungen werden mit Hilfe von <InternalLink>-Variablen dargestellt. Die <InternalLink>-Variablen müssen unter dem gemeinsamen übergeordneten Element der verschiedenen Baugruppenträger (z. B. Gerät) hinzugefügt werden. Der Name des internen Links muss im gemeinsamen übergeordneten Element eindeutig sein.

```
<InternalLink Name="Link To [Internal link Name]" RefPartnerSideA="[Communication Port UniqueID]:[Communication Port External Interface Name]" RefPartnerSideB="[Communication Port UniqueID]:[Communication Port External Interface Name]" />
```

6.5.8 AML-Datei mit benutzerdefinierten GSD/GSDML-Attributen exportieren/importieren

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Benutzerdefinierte Attribute erlauben meist den Austausch von Daten, die für ein bestimmtes Gerät oder Modul gelten, und für die es keine AML-basierte Spezifikation gibt. In TIA Portal V18 werden benutzerdefinierte Attribute für alle Arten von Modulen wie GSD, GSDML sowie für Nicht-GSD/GSDML-basierte Module unterstützt. Nur zusätzliche Daten der Module können mit benutzerdefinierten Attributen ausgetauscht werden, jedoch keine zusätzlichen Daten an Ports, Schnittstellen, Knoten usw. Benutzerdefinierte Attribute sind in AR APC als unsortierte Liste von Namen-Werte-Paaren definiert.

Weitere Informationen zu benutzerdefinierten Nicht-GSD/GSDML-Attributen finden Sie unter AML-Datei mit benutzerdefinierten Nicht-GSD/GSDML-Attributen exportieren/importieren (Seite 1506)

Die AML-Struktur sieht wie folgt aus:

```
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="AttributeName1" AttributeDataType="xs:string">
<Value>AttributeValue1</Value>
</Attribute>
<Attribute Name="AttributeName2" AttributeDataType="xs:string">
<Value>AttributeValue2</Value>
</Attribute>
</Attribute>
```

Hinweis

GSD/GSDML-Module unterstützen nicht die Kanalkonfiguration im TIA Portal. Daher kann der Export/Import benutzerdefinierter Kanalattribute über AML nicht unterstützt werden.

Für die korrekte Identifizierung eines benutzerdefinierten Attributs in TIA Portal muss der Name des Attributs mit den folgenden Definitionen übereinstimmen:

Der Name eines benutzerdefinierten Attributs besteht aus drei Teilen <name>, <attribute_location>, <value_location>.

- **<name>** ist eine Zeichenkette, die nur alphanumerische Zeichen, Zahlen, "_" und "." enthält. Der Name ist üblicherweise der Name des Attributs in der GSD/GSDML-Datei. Er ist bei Attributen im Parameterdatensatz ein optionaler Teil, wird aber in anderen Fällen zwingend verlangt.
- **<attribute_location>** legt das Geräteelement fest, an dem sich das Attribut befindet. Dies ist erforderlich, um die Lage des Attributs festzustellen, wenn es sich nicht direkt am Modul befindet. Es hat die Form #<subslotnumber>#<subsubslotnumber>. Das bedeutet, wenn das Attribut sich am Modul befindet, bleibt <attribute_location> leer; wenn es sich an einem Submodul befindet, steht hier "#<subslotnumber>" und wenn es sich an einem Subsubmodul befindet, steht hier "#<subslotnumber>#<subsubslotnumber>".
- **<value_location>** legt fest, wie auf ein Attribut an einem Geräteelement zugegriffen wird. Ist der Wert Teil eines Parameterdatensatzes, wird er mit den folgenden fünf Elementen identifiziert:
 - **DatasetNumber:** Die Nummer des Datensatzes am Geräteelement.
 - **ByteOffset:** Die Stelle in Byte (beginnend mit 0), an der der Wert im Parameterdatensatz beginnt.
 - **Length:** Die Gesamtlänge des Datensatzes in Byte.
 - **BitOffset:** Die Stelle in Bit (beginnend mit 0), an der der Wert im Startbyte (siehe ByteOffset) beginnt.
 - **BitLength:** Die Gesamtlänge des Werts in Bit.

Beispiel für benutzerdefinierte Attribute:

```
<Attribute name = "CustomAttribute">
<RefSemantic CorrespondingAttributePath="ListType"/>
<Attribute Name="IDTP_No_Unit_DIAG#0-1-0-2-7-1" AttributeDataType="xs:string">
<Value>1<Value>
</Attribute>
<Attribute Name="IDTP_LANG#0-1-1-2-0-4" AttributeDataType="xs:string">
<Value>89<Value>
<Attribute Name="IDTP_D_FREEZE#0-1-1-2-7-1" AttributeDataType="xs:string">
<Value>1<Value>
</Attribute>
</Attribute>
```

In einigen (seltenen) Fällen wird das Attribut explizit (nicht als Teil eines Parameterdatensatzes) in der GSD/GSDML-Datei modelliert. Der Zugriff auf den Wert erfolgt dann über den Attributnamen.

Hinweis

Beim Exportieren exportiert TIA Portal immer die kompletten Datensätze, damit Partneranwendungen vollständig auf die enthaltenen Daten zugreifen können.

```
<Attribute name = "CustomAttribute">
<RefSemantic CorrespondingAttributePath="ListType"/>
<Attribute Name="PrmData#0-1-0-2-0-16" AttributeDataType="xs:string">
<Value>128, 0</Value>
</Attribute>
</Attribute>
```

Eine AML-Datei mit benutzerdefinierten Attributen aus PRM-Daten exportieren/importieren

Ab TIA Portal V17 wird es möglich sein, Parameterdaten von GSD/GSDML-Modulen über einen CAX-Export und -Import auszutauschen. In der AML-Datei werden diese Parameterdaten als benutzerdefinierte Attribute dargestellt.

Export

Das folgende Snippet zeigt das Format, in dem Parameterdaten in der AML-Datei erwartet werden, wenn sie über ein steckbares Modul erhalten werden.

```
...
<InternalElement ID="049f1260-7c97-458e-84bd-12682f943f19" Name="Slave_1">
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string"> <Value>GSD:SI018098.GSD/
DAP</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="PrmData-0-0-39-0-312" AttributeDataType="xs:string">
<Value>39,129,0,0,28,0,128,15,255,255,255,255,255,255,255,255,
255,255,255,255,255,255,15,255,255,255,255, 255,255,255,255,255,255,255,255,255</
Value>
</Attribute>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
```


Das folgende Snippet zeigt das Format von Parameterdaten, wenn diese über ein eingebautes Submodul in TIA Portal verfügbar sind, in der AML-Datei jedoch am steckbaren übergeordneten Modul.

```
<InternalElement ID="8fb83cae-ae30-45d2-9d4c-6db1154af02d" Name="IE-AS-i-LINK">
...
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="PrmData#0-130-0-4-0-32" AttributeDataType="xs:string">
<Value>0,0,131,0</Value>
</Attribute>
</Attribute>
<InternalElement ID="574a55bb-1209-4672-86bd-01ee9085eaf6" Name="DAP 1">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
...
```

- Im Snippet oben hat der Name "PrmData#0-130-0-4-0-32" des benutzerdefinierten Attributs die folgende Bedeutung:
 - PrmData#0: Name des Attributs und Positionsnummer des Submoduls, das es enthält
 - 130: Datensatznummer
 - 0: Byte-Offset
 - 4: Datensatzlänge in Byte
 - 0: Bit-Offset
 - 32: Bitlänge
- Benutzerdefinierte Attribute dürfen in der AML-Datei nie an eingebauten Modulen erscheinen, mit Ausnahme eingebauter Module, die sich direkt unter dem Baugruppenträger befinden.
- Sind für ein eingebautes GSD/GSDML-Modul Parameterdaten vorhanden, muss das benutzerdefinierte Attribut über das unmittelbar übergeordnete steckbare Modul oder ein direkt unter dem Baugruppenträger eingebautes übergeordnetes Modul exportiert werden, je nachdem was zutreffend ist.
- Bei beschnittenen Dateien (aus E-Plan exportierte Dateien) darf das eingebaute Submodul, zum dem das benutzerdefinierte Attribut gehört, in der AML-Datei nicht vorhanden sein.

Import

CAX unterstützt den Import von benutzerdefinierten PrmData-Attributen über GSD/GSDML-Module in den folgenden Fällen:

- Wenn ein steckbares GSD/GSDML-Modul mit benutzerdefinierten Attributen angelegt ist.
- Wenn das eingebaute Submodul in der Datei beschnitten ist und die benutzerdefinierten Attribute über das steckbare übergeordnete Modul bereitgestellt werden.
- Wenn die Datei nicht beschnitten ist, sind eingebaute Submodule vorhanden und die benutzerdefinierten Attribute werden über das steckbare übergeordnete Modul bereitgestellt.

Auch benutzerdefinierte Attribute, die als einzelne Attribute und nicht als vollständiger Parameterdatensatz bereitgestellt werden, werden beim CAX-Import unterstützt. Dieses Format wird bei Dateien erwartet, die aus E-Plan exportiert werden.

Das folgende Snippet zeigt ein solches Beispiel.

```
<InternalElement ID="1cf85c26-cc2a-4413-b91b-e4b55e183762" Name="Slave_1">
...
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="PSR-0-4-26-0-1" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="C4F-0-4-26-1-1" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="GPC-0-4-26-2-1" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="SFC-0-4-26-3-1" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="ACC-0-4-26-4-1" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="CMV3.1-0-4-26-5-1" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="MUPR-0-5-26-0-32" AttributeDataType="xs:string">
<Value>0,0,31,64</Value>
</Attribute>
<Attribute Name="TMR-0-9-26-0-32" AttributeDataType="xs:string">
<Value>7,255,225,236</Value>
</Attribute>
<Attribute Name="TSOLF-0-13-26-0-8" AttributeDataType="xs:string">
<Value>100</Value>
</Attribute>
<Attribute Name="PP-0-25-26-0-8" AttributeDataType="xs:string">
<Value>2</Value>
</Attribute>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
...
```

Im Beispiel oben gehören alle benutzerdefinierten Attribute zu Datensatz Nummer 0. Sie versuchen jedoch, den Wert verschiedener Byte(s) und Bit(s) zu verändern.

- Beim CAX-Import müssen sowohl Integer- als auch Hex-Formate (z. B. 0x67) für den Wert eines benutzerdefinierten Attributs unterstützt werden, jedoch immer nur jeweils ein Format für ein beliebiges Attribut.

Eine AML-Datei mit benutzerdefinierten Attributen, die keine PRM-Daten sind, exportieren/importieren

Ab TIA Portal V17 ist es möglich, alle Attribute von GSD/GSDML-Modulen mit Lese-Schreib-Zugriff über den CAX-Export und -Import auszutauschen. In einer AML-Datei müssen diese Attribute als benutzerdefinierte Attribute dargestellt werden.

Export

Beim CAX-Export werden alle Attribute, die in TIA Portal schreibgeschützt sind, ignoriert; der Teil für benutzerdefinierte Attribute in einer von TIA Portal exportierten AML-Datei darf nur schreibbare Attribute enthalten.

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute in der AML-Datei vorliegen sollten, sie über ein steckbares Modul erhalten werden.

```
...
<InternalElement ID="806532c8-109a-42c6-82e9-84e8ba308aad" Name="cp1604">
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="Author" AttributeDataType="xs:string">
<Value>AuthorValue</Value>
</Attribute>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
```

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute exportiert werden, wenn sie über ein eingebautes Submodul in TIA Portal verfügbar sind, in der AML-Datei jedoch am steckbaren übergeordneten Modul.

```
...
<InternalElement ID="806532c8-109a-42c6-82e9-84e8ba308aad" Name="cp1604">
...
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="Failsafe_FIODBNumber#0" AttributeDataType="xs:string">
<Value>0</Value>
</Attribute>
<Attribute Name="Failsafe_FParameterSignatureIndividualParameters#0"
AttributeDataType="xs:string">
<Value>5</Value>
</Attribute>
</Attribute>
<InternalElement ID="6b94abcd-3fe4-4800-9e18-ea7810d7afed" Name="PS_8Byte">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
```

Im Snippet oben bezeichnet #0 die Positionsnummer des eingebauten untergeordneten Elements, zu dem das Attribut tatsächlich gehört.

Beim CAX-Export werden alle Attribute ignoriert, die bereits im Rahmen der geltenden AR APC Versionsempfehlung exportiert werden.

CAX exportiert keine benutzerdefinierten Attribute über Schnittstellen und Ports sowie über darunter angeordnete Submodule. Die einzige Ausnahme sind Parameterdaten (PrmData).

Import

CAX unterstützt den Import von benutzerdefinierten Attributen über GSD/GSDML-Module in den folgenden Fällen:

- Wenn ein steckbares GSD/GSDML-Modul mit benutzerdefinierten Attributen angelegt ist.
- Wenn das eingebaute Submodul in der Datei beschnitten ist und die benutzerdefinierten Attribute über das steckbare übergeordnete Modul bereitgestellt werden.
- Wenn die Datei nicht beschnitten ist, sind eingebaute Submodule vorhanden und die benutzerdefinierten Attribute werden über das steckbare übergeordnete Modul bereitgestellt.

Beim CAX-Export werden alle Attribute ignoriert, die bereits im Rahmen der geltenden AR APC Versionsempfehlung importiert werden und es wird eine entsprechende Warnmeldung erzeugt.

Hinweis

Der Import benutzerdefinierter Attribute ist möglicherweise eingeschränkt, wenn die Attribute komplexe Abhängigkeitshierarchien mit anderen Attributen aufweisen. Beispiel: Attribut A ist abhängig von Attribut B und Attribut C. Der Import von A funktioniert daher erst nach dem Import von B und C. Dieses Verhalten kann modulabhängig unterschiedlich sein. In diesen Fällen kann der Import fehlschlagen und diese Attribute werden übersprungen (mit einer entsprechenden Benachrichtigung an den Benutzer); der Benutzer muss dann im TIA Portal die benutzerdefinierten Attribute, die nicht importiert/übersprungen wurden, nach dem AML-Import explizit konfigurieren.

Siehe auch

AML-Datei mit benutzerdefinierten Nicht-GSD/GSDML-Attributen exportieren/importieren (Seite 1506)

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.9 AML-Datei mit benutzerdefinierten Nicht-GSD/GSDML-Attributen exportieren/importieren

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Mit dem TIA Portal V18 können Sie alle Nicht-GSD/GSDML-Attribute von Hardware-(Sub-)Modulen und Kanälen mit Openness-Lese-Schreib-Zugriff über den CAx-Export und -Import austauschen. In einer AML-Datei müssen diese Attribute als benutzerdefinierte Attribute dargestellt werden.

Eine AML-Datei mit benutzerdefinierten Modulattributen exportieren/importieren

Export

Der CAx-Export berücksichtigt nur in TIA Portal Openness schreib-/lesbare Attribute und exportiert diese in eine AML-Datei. Im TIA Portal sind die Attribute auf verschiedene Ebenen verteilt. Beispiel: Steckbare (Sub-)Module, eingebaute (Sub-)Module. Unabhängig von den unterschiedlichen Hierarchieebenen muss die AML-Datei beim Export der Konfiguration auf der Ebene steckbarer (Sub-)Module immer benutzerdefinierte Attribute enthalten:

1. Wenn ein steckbares (Sub-)Modul Attribute enthält, müssen diese in der exportierten AML-Datei auf derselben Ebene bleiben.
2. Enthält ein eingebautes (Sub-)Modul Attribute, müssen diese in der exportierten AML-Datei in das jeweilige direkt steckbare übergeordnete Modul verschoben werden.

Die folgende Tabelle enthält das Format des identifizierenden Namens für das benutzerdefinierte Attribut in der AML-Datei:

- TypenIdentifer.AttributeName#PositionNumber, wobei

Attribut	Beschreibung
TypenIdentifer	Normalisierter TypenIdentifer (einschließlich FW-Version) eines steckbaren (Sub-)Moduls, der so formatiert ist, dass Sonderzeichen durch '_' ersetzt werden
AttributeName	Name des Attributs in Openness.
PositionNumber	Gilt nur für Attribute eingebauter (Sub-)Module, die unterhalb ihres steckbaren übergeordneten Moduls in der AML-Datei exportiert werden. Dabei handelt es sich um die Positionsnummer des eingebauten (Sub-)Moduls.

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute in der AML-Datei vorliegen sollten, wenn sie über ein steckbares Modul erhalten werden.

```
<InternalElement ID="298d6b6a-fe70-4edd-9230-39b0ae2238a5" Name="PLC_1">
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.Author" AttributeDataType="xs:string">
<Value>cvdfff</Value>
</Attribute>
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.CentralAlarmManagement"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.ClockMemoryByte" AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.CommunicationMode" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.ConfigurationControl"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
</InternalElement>
```

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute exportiert werden, wenn sie über ein eingebautes Submodul in TIA Portal verfügbar sind, in der AML-Datei jedoch am steckbaren übergeordneten Modul.

```
...
<InternalElement ID="298d6b6a-fe70-4edd-9230-39b0ae2238a5" Name="PLC_1">
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.DisplayDefaultLanguage#3"
AttributeDataType="xs:string">
<Value>0</Value>
</Attribute>
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.DisplayProtection#3"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.DisplayTimeToEnergySavingMode#3"
AttributeDataType="xs:string">
<Value>900</Value>
</Attribute>
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.DisplayTimeToStandbyMode#3"
AttributeDataType="xs:string">
<Value>1800</Value>
</Attribute>
</InternalElement>
```

Im Snippet oben bezeichnet #3 die Positionsnummer des eingebauten untergeordneten Elements, zu dem das Attribut tatsächlich im TIA Portal gehört.

Hinweis

Beim CAX-Export werden alle Attribute ignoriert, die bereits im Rahmen der geltenden AR APC Versionsempfehlung exportiert werden.

Import

CAX unterstützt den Import von benutzerdefinierten Attributen über (Sub-)Module in den folgenden Fällen:

- Wenn ein steckbares (Sub-)Modul mit benutzerdefinierten Attributen angelegt ist.
- Wenn das eingebaute Submodul in der Datei beschnitten ist und die benutzerdefinierten Attribute über das steckbare übergeordnete (Sub-)Modul bereitgestellt werden.
- Wenn die Datei nicht beschnitten ist, sind eingebaute Submodule vorhanden und die benutzerdefinierten Attribute werden über das steckbare übergeordnete (Sub-)Modul bereitgestellt.

Beim CAX-Export werden alle Attribute ignoriert, die bereits im Rahmen der geltenden AR APC-Empfehlung importiert werden und es wird eine entsprechende Warnmeldung erzeugt.

Toleranter Import

Beim Import ist das TIA Portal in Bezug auf Namen benutzerdefinierter Attribute tolerant. Das Präfix "Typenidentifizier." ist optional.

- Der CAX-Import akzeptiert AML-Dateien mit benutzerdefinierten Attributen mit vollständigem IdentifyingName (Format: Typenidentifizier.AttributeName#PositionNumber)
- Der CAX-Import akzeptiert AML-Dateien mit benutzerdefinierten Attributen mit teilweiseem IdentifyingName – nur Attributname und Suffixe (wie #PositionNumber), jedoch ohne Typenidentifizier.

Das folgende Snippet zeigt verschiedene, beim CAX-Import akzeptierte Namensformate.

```
...
<InternalElement ID="298d6b6a-fe70-4edd-9230-39b0ae2238a5" Name="PLC_1">
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7516_3AN03_0AB0_V2_9.CentralAlarmManagement"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute Name="CentralAlarmManagement" AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
</InternalElement>
```


Eine AML-Datei mit benutzerdefinierten Kanalattributen exportieren/importieren

Export

Der CAX-Export berücksichtigt nur in Openness schreib-/lesbare Attribute der Kanäle und exportiert diese in eine AML-Datei. Im TIA Portal sind die Kanäle auf verschiedene Ebenen verteilt. Beispiel: Steckbare (Sub-)Module, eingebaute (Sub-)Module. Die AML-Datei muss immer Kanäle (zusammen mit den Attributen) auf der gleichen Ebene enthalten, wenn die Konfiguration exportiert wird.

Nachfolgend ist das Format des identifizierenden Namens für das benutzerdefinierte Attribut in der AML-Datei dargestellt.

- <Typenidentifizierer>.Kanäle.<Type><IoType>_<ChannelNumber>.<AttributeName>#<PositionNumber>, wobei

Benutzerdefiniertes Attribut	Beschreibung
Typenidentifizierer	Normalisierter Typenidentifizierer (einschließlich FW-Version) eines steckbaren (Sub-)Moduls, der so formatiert ist, dass Sonderzeichen durch '_' ersetzt werden.
AttributeName	Name des Attributs in Openness
PositionNumber	Dies gilt nur für Attribute von Kanälen, wenn die Kanäle unter eingebauten (Sub-)Modulen modelliert werden. Dabei handelt es sich um die Positionsnummer des eingebauten (Sub-)Moduls.
Type	Dies ist der Kanaltyp, analog oder digital (kurz als A oder D bezeichnet)
IoType	Dies ist der IO-Kanaltyp, Eingang oder Ausgang (kurz als I oder O bezeichnet)
ChannelNumber	Dies ist die fortlaufende Integerzahl jedes Kanals, von 0 bis N

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute in der AML-Datei vorliegen sollten, wenn der Kanal über ein steckbares Modul gefunden wird.

```

...
<InternalElement ID="928ac5b5-2625-4f40-a624-d731ed673522" Name="DO 8x24VDC/
0.5A_1"><Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>DO8 x 24VDC / 0.5A</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>5</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 322-8BF00-0AB0</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7322_8BF00_0AB0.Author" AttributeDataType="xs:string">
<Value>Z003NH6D</Value>
</Attribute>
</Attribute>
<ExternalInterface ID="143cf656-2a27-4673-9a6a-55bc570068f6" Name="Channel_DO_0"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Digital</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Output</Value>
</Attribute>
<Attribute Name="Number" AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7322_8BF00_0AB0.Channels.DO_0.DiagnosticsNoSupplyVoltage"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7322_8BF00_0AB0.Channels.DO_0.DiagnosticsShortCircuitToGround"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7322_8BF00_0AB0.Channels.DO_0.DiagnosticsShortCircuitToLplus"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7322_8BF00_0AB0.Channels.DO_0.DiagnosticsWireBreak"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7322_8BF00_0AB0.Channels.DO_0.SubstituteValue"
AttributeDataType="xs:string"><Value>False</Value>
</Attribute>

```

</Attribute>

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute exportiert werden, wenn sie über ein eingebautes Submodul gefunden werden.

```

...
<InternalElementID="5bda81fa-8c31-4e1a-b41a-f5100eb2ff2f"Name="AI5/AQ2_1">
<AttributeName="PositionNumber"AttributeDataType="xs:int">
<Value>8</Value>
</Attribute>
<AttributeName="BuiltIn"AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<AttributeName="Address">
<RefSemanticCorrespondingAttributePath="OrderedListType"/>
<AttributeName="1">
<AttributeName="StartAddress"AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<AttributeName="Length"AttributeDataType="xs:int">
<Value>80</Value>
</Attribute>
<AttributeName="IoType"AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
</Attribute>
<AttributeName="2">
<AttributeName="StartAddress"AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<AttributeName="Length"AttributeDataType="xs:int">
<Value>32</Value>
</Attribute>
<AttributeName="IoType"AttributeDataType="xs:string">
<Value>Output</Value>
</Attribute>
</Attribute>
</Attribute>
<ExternalInterfaceID="76a8ea3f-1da6-4622-
b806-912e1da53980"Name="Channel_AI_0"RefBaseClassPath="AutomationProjectConfigurationInter
faceClassLib/Channel">
<AttributeName="Type"AttributeDataType="xs:string">
<Value>Analog</Value>
</Attribute>
<AttributeName="IoType"AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
<AttributeName="Number"AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<AttributeName="Length"AttributeDataType="xs:int">
<Value>16</Value>
</Attribute>
<AttributeName="CustomAttributes">
<RefSemanticCorrespondingAttributePath="ListType"/>
<AttributeName="6ES7511_1CK00_0AB0_V2_9.Channels.AI_0.HardwareInterruptLowLimit2Active#8"AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<AttributeName="6ES7511_1CK00_0AB0_V2_9.Channels.AI_0.OperatingRange#8"AttributeDataType="

```

```
xs:string">
<Value>OperatingRange.PlusMinus10V</Value>
</Attribute>
<AttributeName="6ES7511_1CK00_0AB0_V2_9.Channels.AI_0.OperatingType#8"AttributeDataType="xs:string">
<Value>OperatingType.Voltage</Value>
</Attribute>
<AttributeName="6ES7511_1CK00_0AB0_V2_9.Channels.AI_0.ParameterSettings#8"AttributeDataType="xs:string">
<Value>ParameterSettings.Manual</Value>
</Attribute>
<AttributeName="6ES7511_1CK00_0AB0_V2_9.Channels.AI_0.Smoothing#8"AttributeDataType="xs:string">
<Value>Smoothing.None</Value>
</Attribute>
</Attribute>
</ExternalInterface>
```

Hinweis

Im Snippet oben bezeichnet #8 die Positionsnummer des eingebauten untergeordneten Elements, zu dem der Kanal tatsächlich im TIA Portal gehört. Beim CAX-Export werden alle Attribute ignoriert, die bereits im Rahmen der geltenden AR APC Versionsempfehlung exportiert werden.

Import

CAX unterstützt den Import von benutzerdefinierten Kanalattributen in den folgenden Fällen:

- Wenn die Datei nicht beschnitten ist und Kanäle mit ihren benutzerdefinierten Attributen unter (Sub-)Modul vorhanden sind.
- Wenn das eingebaute Submodul in der Datei beschnitten ist und die Kanäle mit ihren benutzerdefinierten Attributen über das steckbare übergeordnete (Sub-)Modul bereitgestellt werden.
- Wenn das eingebaute Submodul in der Datei beschnitten ist und die Kanäle unter das steckbare übergeordnete Submodul verschoben und die benutzerdefinierten Kanalattribute unter die benutzerdefinierten Attribute des steckbaren Geräteelements mit entsprechender Kanal-Info und Zielgeräte-Info verschoben werden.

Alle benutzerdefinierten Attribute, die einen Satz Enum-Werte unterstützen, werden mit einem voll qualifizierten Enum-Textwert beim Import voll qualifizierter Enum-Textwerte sowie der entsprechenden Integer-Werte exportiert.

Das folgende Snippet zeigt benutzerdefinierte Kanalattribute.

```
<InternalElement ID="0d5e860b-6581-467a-be7a-fcfa37e0ee6b" Name="DI 32x24VDC HF_1">
...
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 521-1BL00-0AB0</Value>
</Attribute>
<InternalElement ID="198bf5e3-ac19-4465-8e3e-8c2faf6ab217" Name="DI 32x24VDC HF_1">
...
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="ff89bfc5-ee51-4525-a388-a9b02213515e" Name="Channel_DI_0"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
...
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.DiagnosticsNoSupplyVoltage#1"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.DiagnosticsWireBreak#1"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
...
</Attribute>
</ExternalInterface>
...
```



```

<InternalElement ID="0d5e860b-6581-467a-be7a-fcfa37e0ee6b" Name="DI 32x24VDC HF_1">
...
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 521-1BL00-0AB0</Value>
</Attribute>
<ExternalInterface ID="ff89bfc5-ee51-4525-a388-a9b02213515e" Name="Channel_DI_0"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
...
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.DiagnosticsNoSupplyVoltage#1"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.DiagnosticsWireBreak#1"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
...
</Attribute>
</ExternalInterface>
...

```

Toleranter Import

Beim Import ist das TIA Portal in Bezug auf Namen benutzerdefinierter Attribute tolerant. Das Präfix "TypeIdentifier" oder Kanalinformationen sind optional.

Ein benutzerdefiniertes Kanalattribut ist auf dem Kanal verfügbar:

- Der CAX-Import akzeptiert AML-Dateien mit benutzerdefinierten Attributen mit vollständigem IdentifyingName (Format: TypeIdentifier.Channels.TypeIoType_ChannelNumber.AttributeName#PositionNumber)
- Der CAX-Import akzeptiert AML-Dateien mit benutzerdefinierten Attributen mit teilweisem IdentifyingName – nur Attributname und Suffixe (wie #PositionNumber), jedoch ohne TypeIdentifier und Kanalinformationen.
- Die unterstützten Formate sind nachfolgend angegeben:
 - <TypeIdentifier>.Channels.<Type><IoType>_<ChannelNumber>.<AttributeName>#<PositionNumber>
 - <AttributeName>#<PositionNumber>

Das folgende Snippet zeigt verschiedene, beim CAX-Import akzeptierte Namensformate.

```
...
<ExternalInterface ID="ff89bfc5-ee51-4525-a388-a9b02213515e" Name="Channel_DI_0"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
...
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.DiagnosticsNoSupplyVoltage#1"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute Name="Channels.DI_0.DiagnosticsWireBreak#1" AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
...
</Attribute>
</ExternalInterface>
...
```

Ein benutzerdefiniertes Kanalattribut ist auf dem einsteckbaren Geräteelement verfügbar:

- Der CAX-Import akzeptiert AML-Dateien mit benutzerdefinierten Attributen mit vollständigem IdentifyingName (Format: TypelIdentifier.Channels.TypeIoType_ChannelNumber.AttributeName#PositionNumber)
- Der CAX-Import akzeptiert AML-Dateien mit benutzerdefinierten Attributen mit teilweiseem IdentifyingName – nur Attributname mit Kanalinfo (Channels.<Type><IoType>_<ChannelNumber>) und Suffixe (wie #PositionNumber), jedoch ohne TypelIdentifier.
- Die unterstützten Formate sind nachfolgend angegeben:
 - <TypelIdentifier>.Channels.<Type><IoType>_<ChannelNumber>.<AttributeName>#<PositionNumber>
 - Channels.<Type><IoType>_<ChannelNumber>.<AttributeName>#<PositionNumber>

Das folgende Snippet zeigt verschiedene, beim CAX-Import akzeptierte Namensformate.

```
...
<ExternalInterface ID="ff89bfc5-ee51-4525-a388-a9b02213515e" Name="Channel_DI_0"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
...
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.DiagnosticsNoSupplyVoltage#1"
AttributeDataType="xs:string"
<Value>True</Value>
</Attribute>
<Attribute Name="Channels.DI_0.DiagnosticsWireBreak#1" AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
...
</Attribute>
</ExternalInterface>
...
```

Ungültiges benutzerdefiniertes Kanalattribut

CAX unterstützt in den folgenden Fällen nicht den Import benutzerdefinierter Kanalattribute:

- Wenn die benutzerdefinierten Kanalattribute sowohl unter den benutzerdefinierten Attributen für Geräteelemente als auch unter benutzerdefinierten Kanalattributen vorhanden sind, werden nur die benutzerdefinierten Kanalattribute unter dem Geräteelement verarbeitet, die anderen benutzerdefinierten Attribute unter den Kanälen werden ignoriert und eine Warnmeldung erzeugt.

6.5 Hardware-Daten importieren/exportieren

Das folgende Snippet zeigt benutzerdefinierte Kanalattribute unter Geräteelementen und Kanälen.

```
<InternalElement ID="0d5e860b-6581-467a-be7a-fcfa37e0ee6b" Name="DI 32x24VDC HF_1">
..
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 521-1BL00-0AB0</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Author" AttributeDataType="xs:string">
<Value>z003tyvt</Value>
</Attribute>
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.EnableValueStatus" AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.StartupComparisonPresetToActualModule"
AttributeDataType="xs:string">
<Value>0</Value>
</Attribute>
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.DiagnosticsNoSupplyVoltage#1"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.DiagnosticsWireBreak#1"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
</Attribute>
...
<ExternalInterface ID="ff89bfc5-ee51-4525-a388-a9b02213515e" Name="Channel_DI_0"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
...
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute
Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.HardwareInterruptFallingEdgeActive#1"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute
Name="6ES7521_1BL00_0AB0_V2_2.Channels.DI_0.HardwareInterruptFallingEdgeEventName#1"
AttributeDataType="xs:string">
<Value>Falling edge0</Value>
</Attribute>
</ExternalInterface>
...
```

Hinweis

Der Import benutzerdefinierter Attribute ist möglicherweise eingeschränkt, wenn die Attribute komplexe Abhängigkeitshierarchien mit anderen Attributen aufweisen. Beispiel: Attribut A ist abhängig von Attribut B und Attribut C. Der Import von A funktioniert daher erst nach dem Import von B und C. Dieses Verhalten kann modulabhängig unterschiedlich sein. In diesen Fällen kann der Import fehlschlagen und diese Attribute werden übersprungen (mit einer entsprechenden Benachrichtigung an den Benutzer); der Benutzer muss dann im TIA Portal die benutzerdefinierten Attribute, die nicht importiert/übersprungen wurden, nach dem AML-Import explizit konfigurieren.

AML-Datei mit benutzerdefinierten Attributen mit Netzwerkobjekten exportieren/importieren

CAX exportiert/importiert benutzerdefinierte Lese- und Schreibattribute für verschiedene Netzwerkobjekte wie Schnittstelle, Port, Knoten, IO-System und Subnetz.

NetworkInterface-Konfiguration

Die folgende Tabelle enthält das Format des identifizierenden Namens für das benutzerdefinierte Schnittstellenattribut in der AML-Datei.

- <Typidentifizier>.Schnittstelle.<Label>.<AttributeName>, wobei

Benutzerdefiniertes Attribut	Beschreibung
Typidentifizier	Normalisierter Typidentifizier (einschließlich FW-Version) eines steckbaren (Sub-)Moduls, der so formatiert ist, dass Sonderzeichen durch '_' ersetzt werden.
AttributeName	Name des Attributs in Openness

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute in der AML-Datei vorliegen sollten, wenn die Schnittstelle über ein steckbares Modul gefunden wird.

```
...
<InternalElement ID="d539b22a-9ff4-4cf7-b0f4-2f46c652f82a" Name="PROFINET interface_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>32768</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute
Name="6DL4158_3FH04_3XX0_V4_1_S74100.Interfaces.X1.DeviceReplacementWithoutExchangeableMed
ium" AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute
Name="6DL4158_3FH04_3XX0_V4_1_S74100.Interfaces.X1.DiagnosticsCommunicationError"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="6DL4158_3FH04_3XX0_V4_1_S74100.Interfaces.X1.DisplayUpdateInterval"
AttributeDataType="xs:string">
<Value>DisplayUpdateInterval.Value10Seconds</Value>
</Attribute>
<Attribute Name="6DL4158_3FH04_3XX0_V4_1_S74100.Interfaces.X1.IECV22LLDPMode"
AttributeDataType="xs:string"><Value>True</Value>
</Attribute>
<Attribute Name="6DL4158_3FH04_3XX0_V4_1_S74100.Interfaces.X1.KeepAlivesInterval"
AttributeDataType="xs:string"><Value>30</Value>
</Attribute>
<Attribute Name="6DL4158_3FH04_3XX0_V4_1_S74100.Interfaces.X1.PnSendClock"
AttributeDataType="xs:string"><Value>1000000</Value>
</Attribute>
<Attribute Name="6DL4158_3FH04_3XX0_V4_1_S74100.Interfaces.X1.TimeSynchronizationNtp"
AttributeDataType="xs:string"
<Value>True</Value>
</Attribute>
```

Portkonfiguration

Die folgende Tabelle enthält das Format des identifizierenden Namens für das benutzerdefinierte Attribut Port in der AML-Datei.

- <TypIdentifizier>.Schnittstellen.<InterfaceLabel>.Ports.<PortLabel>.<AttributeName>, wobei

Benutzerdefiniertes Attribut	Beschreibung
Typidentifizier	Normalisierter Typidentifizier (einschließlich FW-Version) eines steckbaren (Sub-)Moduls, der so formatiert ist, dass Sonderzeichen durch '_' ersetzt werden.
AttributeName	Name des Attributs in Openness

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute in der AML-Datei vorliegen sollten, wenn der Port über eine Schnittstelle gefunden wird.

```
...
<InternalElement ID="fc31e87d-2162-43f6-ae4c-211fb5e21dec" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P1R</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>32769</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Ports.P1R.AlternativePartnerPorts"
AttributeDataType="xs:string"><Value>False</Value>
</Attribute>
<Attribute
Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Ports.P1R.EndOfDetectionOfAccessibleDevices"
AttributeDataType="xs:string"><Value>False</Value>
</Attribute>
<Attribute Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Ports.P1R.EndOfSyncDomain"
AttributeDataType="xs:string"
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Ports.P1R.EndOfTopologyDiscovery"
AttributeDataType="xs:string"
<Value>False</Value>
</Attribute>
<Attribute Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Ports.P1R.PortActivation"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Ports.P1R.PortMonitoring"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute
Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Ports.P1R.TransmissionRateAndDuplex"
AttributeDataType="xs:string">
<Value>TransmissionRateAndDuplex.Automatic</Value>
</Attribute>
```

Knotenkonfiguration

Die folgende Tabelle enthält das Format des identifizierenden Namens für das benutzerdefinierte Attribut Knoten in der AML-Datei.

- <Typeldentifizier>.Schnittstellen.<InterfaceLabel>.Knoten.<NodeLabel>.<AttributeName>, wobei

Benutzerdefiniertes Attribut	Beschreibung
Typeldentifizier	Normalisierter Typeldentifizier (einschließlich FW-Version) eines steckbaren (Sub-)Moduls, der so formatiert ist, dass Sonderzeichen durch '_' ersetzt werden.
AttributeName	Name des Attributs in Openness

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute in der AML-Datei vorliegen sollten, wenn der Knoten über eine Schnittstelle gefunden wird.

```
...
<InternalElement ID="609cefa0-208f-4c51-bee5-60002332ef84" Name="E1">
<Attribute Name="SubnetMask" AttributeDataType="xs:string">
<Value>255.255.255.0</Value>
</Attribute>
<Attribute Name="IpProtocolSelection" AttributeDataType="xs:string">
<Value>Project</Value>
</Attribute>
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Ethernet</Value>
</Attribute>
<Attribute Name="NetworkAddress" AttributeDataType="xs:string">
<Value>192.168.0.1</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute
Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Nodes.Ethernet.PnDeviceNameAutoGeneration"
AttributeDataType="xs:string"><Value>True</Value>
</Attribute>
<Attribute
Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Nodes.Ethernet.PnDeviceNameSetDirectly"
AttributeDataType="xs:string"><Value>False</Value>
</Attribute>
<Attribute Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Nodes.Ethernet.UseIsoProtocol"
AttributeDataType="xs:string"><Value>False</Value>
</Attribute>
<Attribute Name="6ES7511_1AK00_0AB0_V1_8.Interfaces.X1.Nodes.Ethernet.UseRouter"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationEthernetRoleClassLib/
NodeEthernet" />
</InternalElement>
```


Subnetzkonfiguration

Die folgende Tabelle enthält das Format des identifizierenden Namens für das benutzerdefinierte Attribut Subnetz in der AML-Datei.

- <Typenidentifizierer>.<Attributname>, wobei

Benutzerdefiniertes Attribut	Beschreibung
Typenidentifizierer	Normalisierter Typenidentifizierer eines Subnetzes, der so formatiert ist, dass Sonderzeichen durch '_' ersetzt werden.
Attributname	Name des Attributs in Openness

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute in der AML-Datei vorliegen sollten, wenn das Subnetz gefunden wird.

```
...
<InternalElement ID="cb471149-05a1-4063-b591-25a00d84c288" Name="PROFIBUS_1">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Profibus</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="System_Subnet_Profibus.BusProfile" AttributeDataType="xs:string">
<Value>BusProfile.Dp</Value>
</Attribute>
<Attribute Name="System_Subnet_Profibus.HighestAddress" AttributeDataType="xs:string">
<Value>126</Value>
</Attribute>
<Attribute Name="System_Subnet_Profibus.PbCableConfiguration"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute Name="System_Subnet_Profibus.PbCyclicDistribution"
AttributeDataType="xs:string">
<Value>True</Value>
</Attribute>
<Attribute Name="System_Subnet_Profibus.SubnetId" AttributeDataType="xs:string">
<Value>A5B5-1</Value>
</Attribute>
<Attribute Name="System_Subnet_Profibus.TransmissionSpeed"
AttributeDataType="xs:string"><Value>BaudRate.Baud1500000</Value>
</Attribute>
</Attribute>
```

IO-Systemkonfiguration

Die folgende Tabelle enthält das Format des identifizierenden Namens für das benutzerdefinierte Attribut IO-System in der AML-Datei.

- <Typenidentifizierer>.Schnittstellen.<Schnittstellenlabel>.IOSystem.<Attributname>, wobei

Benutzerdefiniertes Attribut	Beschreibung
Typidentifizier	Normalisierter Typidentifizier (einschließlich FW-Version) eines steckbaren (Sub-)Moduls, der so formatiert ist, dass Sonderzeichen durch '_' ersetzt werden.
AttributeName	Name des Attributs in Openness

Das folgende Snippet zeigt das Format, in dem benutzerdefinierte Attribute in der AML-Datei vorliegen sollten, wenn das IO-System über eine Schnittstelle gefunden wird.

```
...
<InternalElement ID="f6edc86b-f995-4459-9429-b30c36b0b475" Name="PROFINET IO-System">
<Attribute Name="Number" AttributeDataType="xs:int">
<Value>100</Value>
</Attribute>
<Attribute Name="CustomAttributes">
<RefSemantic CorrespondingAttributePath="ListType" />
<Attribute Name="6ES7515_2AM00_0AB0_V1_8.Interfaces.X1.IoSystem.MultipleUseIoSystem"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
<Attribute
Name="6ES7515_2AM00_0AB0_V1_8.Interfaces.X1.IoSystem.UseIoSystemNameAsDeviceNameExtension"
AttributeDataType="xs:string">
<Value>False</Value>
</Attribute>
</Attribute>
```

Toleranter Import

Beim Import ist das TIA Portal in Bezug auf Namen benutzerdefinierter Attribute in den folgenden Szenarien tolerant:

- Der CAX-Import akzeptiert AML-Dateien mit benutzerdefinierten Attributen mit vollständigem IdentifyingName.
- Der CAX-Import akzeptiert AML-Dateien mit benutzerdefinierten Attributen mit ausschließlich dem Attributnamen.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.10 AML-Datei mit einsteckbarer Portkonfiguration exportieren/importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet
Siehe Projekt öffnen (Seite 131)

Einleitung

Im TIA Portal können Geräte über eingebaute und steckbare Portkonfigurationen verfügen. Während eingebaute Ports in AML-Dateien ohne Änderungen dargestellt werden können, ist für steckbare Ports eine Umwandlung erforderlich, da externe Tools wie EPLAN die Konfiguration/den Austausch von steckbaren Ports direkt unter einer Netzwerkschnittstelle nicht unterstützen.

Daher wurde mit AR APC 1.4.0 ein Umwandlungsansatz eingeführt, der den ursprünglichen steckbaren Port in zwei Teile (Pluggable DeviceItem und BuiltIn CommunicationPort) innerhalb der AML-Datei aufteilt. Diese Teile werden dann über einen neuen externen Schnittstellentyp und eine interne Verbindung verbunden.

Die benutzerdefinierten Attribute bleiben nach dieser Umwandlung unter dem steckbaren DeviceItem zugänglich.

TIA Portal und AML-Hierarchie

TIA Portal Hierarchy

```
-> GSD device_1
  -> Rack
    -> ET200SP
      ->IM 155-6 PN/3 HF V4.2
        ->PN-IO
          ->IE1
            ->Port 1 (SCRJ/RJ45)(pluggable port)
            ->Port 2 (LC/FC)(pluggable port)
            ->Port 3 (2xFC)(pluggable port)
```

AML File Hierarchy

```
-> GSD device_1
  -> Rack
    -> ET200SP
      ->IM 155-6 PN/3 HF V4.2
        ->PN-IO
          ->IE1
            ->Port 1 (SCRJ/RJ45)(built-In CommunicationPort)
            ->Port 2 (LC/FC)(built-In CommunicationPort)
            ->Port 3 (2xFC)(built-In CommunicationPort)
          ->Port 1 (SCRJ/RJ45)(Pluggable DeviceItem which is connected to Port 1 of IE1)
          ->Port 2 (LC/FC)(Pluggable DeviceItem which is connected to Port 2 of IE1)
          ->Port 3 (2xFC)(Pluggable DeviceItem which is connected to Port 3 of IE1)
```

Export

Die AML-Datei, die beim Export für obige Konfiguration erzeugt wird, ist nachfolgend dargestellt:

```

<?xmlversion="1.0"encoding="utf-8"?>
<CAEXFilexmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"FileName="PluggablePortTransformations.aml"SchemaVersion="2.15"xsi:noNamespaceSch
emaLocation="CAEX_ClassModel_V2.15.xsd">.
/..
<InstanceHierarchyName="APCSampleInstanceHierarchy"><InternalElementID="7b00480c-5708-4ffa
-ac2d-d571b57be6af"Name="GSDdevice_1">
<AttributeName="TypeIdentifier"AttributeDataType="xs:string">
<Value>GSD:GSDML-V2.33-SIEMENS-ET200SP-20180406.XML/D</Value>
</Attribute></InternalElement>
<InternalElementID="63dbb1a0-3b53-47b9-a38e-333b15a93f59"Name="Port1 (SCRJ/RJ45) ">
<AttributeName="Label"AttributeDataType="xs:string"><Value>P1R</Value>
</Attribute>
<AttributeName="PositionNumber"AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<AttributeName="BuiltIn"AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterfaceID="c3041983-aad3-41b6-a378-
c8c2f5dc2608"Name="CommunicationPortInterface"RefBaseClassPath="AutomationProjectConfigura
tionInterfaceClassLib/CommunicationPortInterface"/>
<ExternalInterfaceID="c738690b-a613-4bcb-bba7-
ed1c35c0f5a9"Name="CommunicationPortProxyInterface"RefBaseClassPath="AutomationProjectConf
igurationInterfaceClassLib/CommunicationPortProxyInterface"/
><SupportedRoleClassRefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort"/>
</InternalElement>
</InternalElement>
<InternalElementID="63ac85bc-b31f-4176-b9dd-894366458d72"Name="Port1 (SCRJ/RJ45) ">
<AttributeName="TypeName"AttributeDataType="xs:string">
<Value>Port1 (SCRJ/RJ45) </Value>
</Attribute>
<AttributeName="PositionNumber"AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<AttributeName="BuiltIn"AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<AttributeName="TypeIdentifier"AttributeDataType="xs:string">
<Value>GSD:GSDML-V2.33-SIEMENS-ET200SP-20180406.XML/SM/IDS_1P1HFFO_RJ45V4.3</Value>
</Attribute>
<AttributeName="InstallationDate"AttributeDataType="xs:dateTime">
<Value>2023-07-25T11:10:49.3048765Z</Value>
</Attribute>
<AttributeName="CustomAttributes">
<RefSemanticCorrespondingAttributePath="ListType"/
><AttributeName="GSD_GSDML_V2_33_SIEMENS_ET200SP_20180406_XML_SM_IDS_1P1_HF_FO_RJ45_V4_3.A
lternativePartnerPorts"AttributeDataType="xs:string">
<Value>False</Value></Attribute>
...
</Attribute>
<ExternalInterfaceID="0655ee61-7a4b-4ece-99a9cb626893f996"Name="CommunicationPortProxyInte
rface"RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortProxyInterface"/

```

```

><SupportedRoleClassRefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem"/>
</InternalElement>
</InternalElement>
<SupportedRoleClassRefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem"/>
<InternalLinkName="LinkToProxyPort_1"RefPartnerSideA="63ac85bc-b31f-4176-
b9dd-894366458d72:CommunicationPortProxyInterface"RefPartnerSideB="63dbb1a0-3b53-47b9-
a38e-333b15a93f59:CommunicationPortProxyInterface"/>
<InternalLinkName="LinkToProxyPort_2"RefPartnerSideA="156250d0-f6f6-4bc7-97c6-
e910455c78ae:CommunicationPortProxyInterface"RefPartnerSideB="79183be2-9768-4f32-
a2ff-9ece9f65a64e:CommunicationPortProxyInterface"/>
</InternalElement>
<SupportedRoleClassRefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem"/>
</InternalElement>
<SupportedRoleClassRefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Device"/>
</InternalElement>
<SupportedRoleClassRefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceUserFolder"/>
</InternalElement>
<SupportedRoleClassRefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
AutomationProject"/>
</InternalElement>
</InstanceHierarchy>
</CAEXFile>

```

Port- und DeviceItem-Verbindung

Die XML-Darstellung von Port-DeviceItem-Verbindungen zwischen mehreren Ports und DeviceItems erfolgt im unten dargestellten Format.

- ExternalInterface-<ExternalInterface> des Typs "CommunicationPortProxyInterface" wird unter den internen Elementen <CommunicationPort> und <DeviceItem> hinzugefügt, die Teil der Verbindung sind.

```

<InternalElement ID="[Port Unique ID]" Name="[Port Name]">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>PlR</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="[External Interface Unique ID]"
Name="CommunicationPortProxyInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortProxyInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="[DeviceItem Unique ID]" Name="[DeviceItem Name]">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>Port 2 (LC/FC)</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>GSD:GSDML-V2.33-SIEMENS-ET200SP-20180406.XML/SM/IDS_1P2 HF LC_FC V4.3</Value>
</Attribute>
<ExternalInterface ID="c7066655-8895-4969-ae44-4533ad5319d7"
Name="CommunicationPortProxyInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortProxyInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>

```

- InternalLink-Port-DeviceItem-Verbindungen werden mithilfe von <InternalLink>-Variablen dargestellt. <InternalLink>-Variablen müssen unter dem gemeinsamen übergeordneten Element der verschiedenen Port- und DeviceItem-Elemente (z. B. Baugruppenträger) hinzugefügt werden. Der Name des internen Links muss im gemeinsamen übergeordneten Element eindeutig sein.

```

<InternalLink Name="Link To [Internal link Name]" RefPartnerSideA="[Communication Port
UniqueID]:CommunicationPortProxyInterface" RefPartnerSideB="[DeviceItem
UniqueID]:CommunicationPortProxyInterface" />

```

Import

Es ist möglich, umgewandelte steckbare Porteinzelheiten aus einer AML-Datei zu importieren, die beim oben genannten Export erzeugt wurde. Es wird jedoch auch möglich sein, AML-Dateien zu importieren, die in früheren Versionen eines TIA Portal-Projekts erzeugt wurden.

Hinweis

- Das veränderte Verhalten der Exporthierarchie gilt erst ab Version V19. Ältere Versionen von TIA Portal behalten das gleiche Verhalten wie zuvor.
 - Die Hierarchie in der AML-Datei beeinflusst/beeinträchtigt die interne Hierarchie in TIA Portal nach dem Import nicht.
 - Dieses Verhalten in Bezug auf die Änderung/Umwandlung der Hierarchie gilt für alle steckbaren Portkonfigurationen (Scalance-Geräte, ET200SP,...).
-

6.5.11 AML-Datei mit IO-Link exportieren/importieren

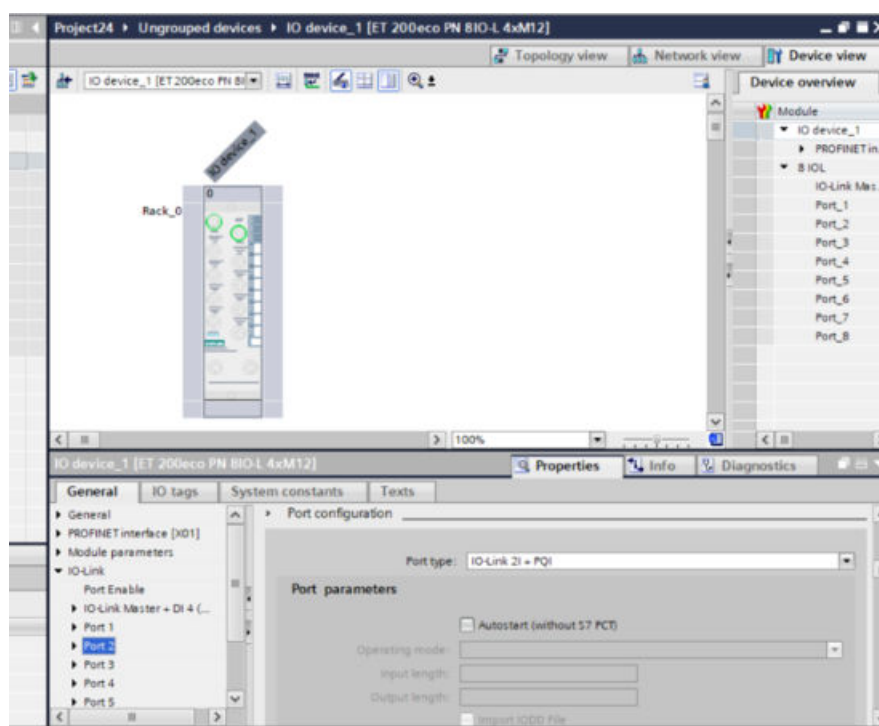
Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

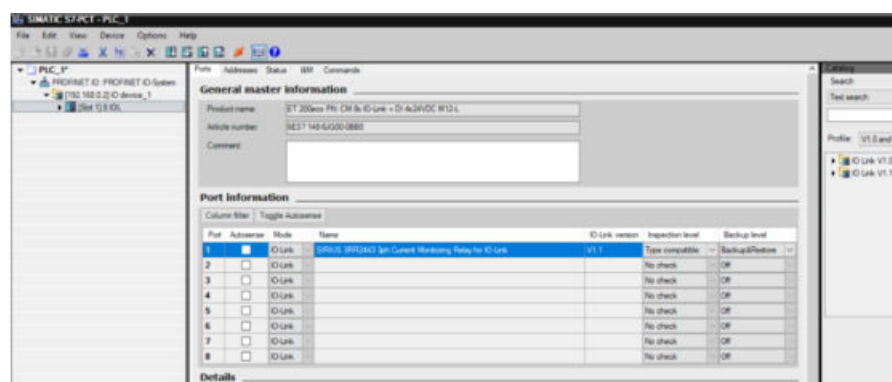
Einleitung

Sie können mit TIA Portal eine AML-Datei mit der IO-Link-Masterkonfiguration zusammen mit IO-Link-Geräten, die über S7-PCT (Port Configuration Tool) konfiguriert wurden, importieren und exportieren.

Nachfolgend ist ein Konfigurationsbeispiel mit einem IO-Link-Master dargestellt.



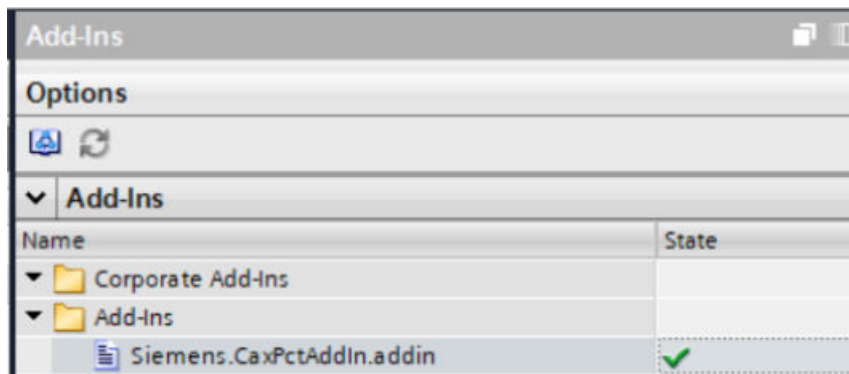
Nachfolgend ist ein Konfigurationsbeispiel für ein IO-Link-Gerät dargestellt, das am IO-Master-Port über PCT konfiguriert wurde:



Add-In aktivieren/deaktivieren

Für den Export/Import der IO-Link-Masterkonfiguration (mit PCT in TIA Portal konfiguriert) über eine AML-Datei muss Siemens.CaxPctAddIn.addin über die Benutzeroberfläche von TIA Portal aktiviert werden.

Das Add-In ist über die Benutzeroberfläche zu aktivieren/deaktivieren. Um das Add-in zu aktivieren, werden Sie aufgefordert, ihre allgemeinen und für den Workflow geltenden Berechtigungen anzugeben.

**Hinweis**

Die Datei 'Siemens.CaxPctAddIn.addin' muss aus "Setup\\Bundles\\WinCCCA_PRO_DVD_2\\DVD\\Support\\CAX\\Add-Ins" kopiert werden, um das Add-In zu aktivieren/deaktivieren.

AML-Datei mit IO-Link-Informationen exportieren

Sie können mit TIA Portal eine Projektkonfiguration mit IO-Link-Mastern zusammen mit den angeschlossenen IO-Link-Geräten, die über PCT konfiguriert wurden, exportieren. Nach einem erfolgreichen Export muss die gleiche Konfiguration in der AML-Datei angezeigt werden. Wird das Add-in nicht aktiviert, ist der Export immer noch erfolgreich möglich, aber die Konfiguration für IO-Link-Geräte wird nicht exportiert.

Bei Problemen mit dem Export PCT-relevanter Konfigurationen werden die Details als Fehler-/Warnmeldung in der PCT.log-Datei protokolliert. Unabhängig davon, ob der PCT-Export erfolgreich ist oder nicht, wird in Register Info der Benutzeroberfläche von TIA Portal eine allgemeine Meldung angezeigt.

Speicherort der PCT-Protokolldatei: Die PCT-Protokolldatei wird am gleichen Ort wie die CAX-Protokolldatei erstellt, d.h. <CAX log location>\\<guid>\\PCT.log.

- Wird der IO-Link-Master ohne S7 PCT konfiguriert (Beispiel: Ports mit Konfiguration ohne S7-PCT = wahr im TIA Portal), weil die IO-Link-Ports über TIA Portal konfiguriert wurden, werden die Ports trotzdem von TIA Portal exportiert, jedoch mit Mindestattributen wie ConfigurationWithPDCT, Label und Name in der AML-Datei, und diese werden unverändert in die AML-Datei zurückimportiert.
- Einige Konfigurationen (wie HW-Parameter/Attribute) im TIA Portal können die IO-Link-Portkonfiguration auf der PCT-Seite beeinflussen. Beispiel: Im Modul 6ES7 148-6JG00-0BB0 kann jeder Port mit einem "Porttyp" in TIA Portal, der den Typ der IO-Link-Geräte beeinflusst, konfiguriert und auf der PCT-Seite in diesen Port gesteckt werden. Dieser Parameter steuert die erforderliche Adressraumzuweisung für jeden Port auf der PCT-Seite für den Anschluss des erforderlichen IO-Link-Geräts. Beim AML-Austausch sollten Sie sicherstellen, dass eine derartige 'beeinflussende' Konfiguration in der AML-Datei eingeschlossen wird, damit der AML-Dateiaustausch erfolgreich ausgeführt werden kann. Fehlt eine solche 'beeinflussende' Konfiguration in der AML-Datei, schlägt der AML-Austausch fehl. Im Beispiel oben kann der Benutzer den Export benutzerdefinierter Attribute (über die CAX-Einstellungen im TIA Portal) einschließen, damit der Parameter 'Porttyp' beim Export aus dem TIA Portal in der AML enthalten ist.

- Es wird grundsätzlich erwartet, dass der Benutzer Konfigurationsänderungen im IO-Link in TIA Portal manuell synchronisiert, indem er das S7-PCT-Tool explizit startet. Daher obliegt es dem Benutzer, auf die Aktualität der IO-Link-Konfiguration vor dem Exportieren der AML-Datei zu achten. Beispiel: Der Benutzer erstellt zuerst eine IO-Link-Konfiguration durch Auswählen der Standardparameter für die Ports im TIA Portal und konfiguriert dann die IO-Link-Geräte mit S7-PCT. Später entschließt er sich, Parameter für die Ports im TIA Portal zu ändern. Sobald er Änderungen vornimmt, muss der Benutzer sicherstellen, dass das S7-PCT-Tool gestartet wird, damit eventuelle Inkonsistenzen auf Grund der Parameteränderung (zum Beispiel dass gesteckte IO-Link-Geräte nicht mehr zum Port passen) mitgeteilt und vom Benutzer vor dem AML-Export beseitigt werden können. Andernfalls kann eine inkonsistente Konfiguration exportiert werden.
- Es gibt Konfigurationen, bei denen die IO-Link-Master-Ports zunächst ohne S7-PCT konfiguriert werden können (z. B.: Einstellung Betriebsart = IO-Link manuell), die aber später im S7-PCT Tool geändert/überschrieben werden können. Diese Konfiguration wird vom AML-Austausch nicht unterstützt.

Das folgende Snippet entspricht einer AML, die nach dem Export der Konfiguration über CAX-Export erstellt wurde:

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
www.w3.org/2001/XMLSchema" xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd"
FileName="Project2_.aml" SchemaVersion="2.15">
  ..
</AdditionalInformation>
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
<InternalElement ID="02afa376-7fbd-46d8-9659-b82959617326" Name="Project24">
  ..
<InternalElement ID="41fec27b-a0fa-4df6-a604-d60008e34d81" Name="Ungrouped devices">
<InternalElement ID="968336bc-bda8-4da6-b3fa-2cc8e8493806" Name="ET 200eco station_1">
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>System:Device.ET200eco</Value>
</Attribute>
<InternalElement ID="41196a85-aaae-42e0-994f-4a46e773af0f" Name="Rack_0">
  ...
<InternalElement ID="165b1850-bba5-4509-af92-acafd8f4f1d1" Name="IO device_1">
  ...
<InternalElement ID="555d1dd7-ec4d-40e6-b34d-06443a255741" Name="IOLink">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>IO-Link</Value>
</Attribute>
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>IO-Link</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<InternalElement ID="4670a2aa-f441-458d-8dcc-e31b7712e325" Name="IOLink Port 1">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>Port 1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute><Attribute Name="ConfigurationWithPDCT" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
  ..
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" /><SupportedRoleClass
RefRoleClassPath="AutomationProjectConfigurationIOLinkRoleClassLib/
CommunicationPortIOLink" />
</InternalElement>
  ..
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" /></InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem">
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationIOLinkRoleClassLib/
DeviceItemIOLinkMaster" />
</InternalElement>
  ..

```

6.5 Hardware-Daten importieren/exportieren

```

</InternalElement>
<InternalElement ID="73292b01-7f22-44c8-98b1-2a007580cea0" Name="SIRIUS 3RR2441 3ph
Current Monitoring Relay for IO-Link">
..
<InternalElement ID="d55d0b10-0d2f-44e6-9b96-72725a88ca41" Name="SIRIUS 3RR2441 3ph
Current Monitoring Relay for IO-Link">
..
<InternalElement ID="ab97d538-5e44-4a97-bab1-d228c1af8c0b" Name="IOLink">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>IO-Link</Value>
</Attribute>
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>IO-Link</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<InternalElement ID="f6ff4404-58fb-43ce-bf5d-e365e55c17b5" Name="IOLink Port 1">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>Port 1</Value>
</Attribute>
<ExternalInterface ID="065e309d-01cc-410e-951d-0fa7cac979d0"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" /
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationIOLinkRoleClassLib/
CommunicationPortIOLink" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" /
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationIOLinkRoleClassLib/
DeviceItemIOLinkDevice" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
Device" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
AutomationProject" />
..
<InternalLink Name="IOLink Port 1" RefPartnerSideA="f6ff4404-58fb-43ce-bf5d-
e365e55c17b5:CommunicationPortInterface" RefPartnerSideB="4670a2aa-f441-458d-8dcc-
e31b7712e325:CommunicationPortInterface" />
</InternalElement>
</InstanceHierarchy>
</CAEXFile>

```

AML-Datei mit IO-Link-Informationen importieren

Sie können die AML-Datei mit IO-Link-Master und der IO-Link-Gerätekonfiguration, die über PCT konfiguriert werden kann, importieren. Nach dem erfolgreichen Import sehen Sie die mit den IO-Link-Masterports verbundenen IO-Link-Geräte, nachdem das Tool PCT geöffnet wurde. Wird das Add-in nicht aktiviert, ist der Import immer noch erfolgreich möglich, aber die Konfiguration für IO-Link-Geräte wird nicht importiert.

Bei Problemen mit dem Import PCT-relevanter Konfigurationen werden die Details als Fehler-/Warnmeldung in der PCT.log-Datei protokolliert. Unabhängig davon, ob der PCT-Import erfolgreich ist oder nicht, wird in Register Info der Benutzeroberfläche von TIA Portal eine allgemeine Meldung angezeigt.

Whitelist

Die folgenden Module unterstützen den Export/Import der IO-Link-Masterkonfiguration.

	IO-Link-Master	Firmware-Version
ET 200SP	6ES7-137-6BD00-0BA0	2.2
ET 200AL	6ES7-147-5JD00-0BA0	1.2
	6ES7-147-5JD00-0BA0	1.1
ET 200pro	6ES7-147-4JD00-0AB0	1.1
ET 200MP	6ES7-547-1JF00-0AB0	1.0
ET 200eco PN	6ES7-148-6JD00-0AB0	1.1
	6ES7-148-6JD00-0AB0	1.0
	6ES7-148-6JG00-0BB0	5.1
	6ES7-148-6JG00-0BB0	1.1
	6ES7-148-6JG00-0BB0	1.0
	6ES7 148-6JE00-0BB0	5.1
	6ES7 148-6JJ00-0BB0	5.1
SIPLUS ET 200SP	6AG1-137-6BD00-2BA0	2.2
	6AG2-137-6BD00-1BA0	2.2

Blacklist

Die folgenden Module unterstützen den Export/Import der IO-Link-Masterkonfiguration nicht.

	IO-Link-Master	Firmware-Version
ET 200S	6ES7-138-4GA50-0AB0	
	3RK1-005-0LB00-0AA0	1.0
ET 200SP	6ES7-137-6BD00-0BA0	2.1
	6ES7-137-6BD00-0BA0	2.0
	6ES7-137-6BD00-0BA0	1.0
ET 200AL	6ES7-147-5JD00-0BA0	1.0
ET 200pro	6ES7-147-4JD00-0AB0	1.0
ET 200eco PN	6ES7-148-6JA00-0AB0	7.0
	6ES7-148-6JA00-0AB0	6.1

	IO-Link-Master	Firmware-Version
S7-1200	6ES7-278-4BD32-0XB0	2.1
	6ES7-278-4BD32-0XB0	2.0
SIPLUS S7-1200	6AG1-278-4BD32-2XB0	2.1
	6AG1-278-4BD32-2XB0	2.0
	6AG1-278-4BD32-4XB0	2.1
	6AG1-278-4BD32-4XB0	2.0
SIPLUS ET 200SP	6AG1-137-6BD00-2BA0	2.1
	6AG2-137-6BD00-1BA0	2.1

Siehe auch

Projekt öffnen (Seite 131)

6.5.12 Verwalten von Verbindungen bei Erweiterungsbaugruppenträgern**Voraussetzung**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

Sie können mithilfe von TIA Portal Openness Verbindungen für Erweiterungsbaugruppenträger abrufen, hinzufügen und entfernen. Dadurch können Sie auch die Unterstützung von TIA Portal Openness für die Realisierung von Verbindungen für Erweiterungsbaugruppenträger beim CAX-Export/Import nutzen.

Programmcode

```
ImConnection imConnection = portDeviceItem.GetService<ImConnection>();
imConnection.Connect(partnerport);
imConnection.Disconnect();
imConnection.GetPartnerPort();
var imConnectionOwner = imConnection.OwnedBy;
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.13 AML-Datei mit komplexer Variablenkonfiguration exportieren/importieren

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet
Siehe Projekt öffnen (Seite 131)

Anwendung

Sie können mit TIA Portal eine AML-Datei, die komplexe Variablen mit benutzerdefinierten Datentypen (UDT) enthält, importieren. Im TIA Portal werden komplexe Variablen erstellt, die in der erwähnten Variablentabelle in der AML-Datei vorhanden sind, jedoch wird kein benutzerdefinierter Datentyp (UDT) erstellt. Die benötigten UDT müssen manuell erstellt werden.

Die logische Adresse einer komplexen Variablen wird aus den Subvariablen der komplexen Variablen berechnet. Aus einer Liste aller Subvariablen einer komplexen Variablen wird die Variable mit der niedrigsten möglichen Adresse ausgewählt und die Adresse dieser Variablen wird der übergeordneten komplexen Variablen zugewiesen. Hat eine Subvariable eine ungültige Adresse, wird die Adresse der übergeordneten komplexen Variablen nicht berechnet und auf leer gesetzt.

Bei einem erfolgreichen Import werden alle gültigen komplexen Variablen, die im TIA Portal angelegt wurden, mit den gültigen logischen Adressen angezeigt. Beim Export gibt es keine Änderungen und nur die Variablen werden exportiert.

AML-Datei mit einem benutzerdefinierten Datentyp (UDT)

Das folgende Beispiel bezieht sich auf eine AML-Datei, die eine komplexe Variable mit einem benutzerdefinierten Datentyp (UDT) enthält.

```

<CAEXFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" FileName="singletag.aml"
SchemaVersion="2.15" xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
<AdditionalInformation>
...
</AdditionalInformation>
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
<InternalElement ID="debc8e5c-2cbb-4fa9-afaa-de242e706dff" Name="Project97">
<InternalElement ID="dc142279-d745-432b-8451-4b282b55b3e5" Name="S71500/ET200MP
station_2">
...
<InternalElement ID="eedd64da-22f0-4b6e-babe-75e09e5cfc46" Name="Rail_0">
...
<InternalElement ID="a6e21bd8-f58c-4920-89ee-588d0eb34645" Name="PLC_2">
...
<InternalElement Name="TagTable" ID="065F26B8-E5BA-4BF3-A210-4E26E19F8A30">
<Attribute Name="AssignToDefault" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<InternalElement Name="M1" ID="4AF292AA-DF8E-4A7A-AC9F-005B4DED47A5">
<Attribute Name="DataType" AttributeDataType="xs:string">
<Value>StructDrive</Value>
</Attribute>
<ExternalInterface Name="On"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Tag"
ID="5FCB82C1-019C-4833-A1DF-16721A5239FC">
<Attribute Name="DataType" AttributeDataType="xs:string">
<Value>BOOL</Value>
</Attribute>
<Attribute Name="LogicalAddress" AttributeDataType="xs:string">
<Value>I0.0</Value>
</Attribute>
<Attribute Name="Comment" AttributeDataType="xs:string">
<Value>Motor drive on</Value>
<Attribute Name="aml-lang=de-DE" AttributeDataType="xs:string">
<Value>Motor drive on</Value>
</Attribute>
</Attribute>
</ExternalInterface>
<ExternalInterface Name="Overheat"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Tag"
ID="331659A8-147B-407A-9629-5D85AE04890D">
<Attribute Name="DataType" AttributeDataType="xs:string">
<Value>BOOL</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
<Attribute Name="LogicalAddress" AttributeDataType="xs:string">
<Value>E0.1</Value>
</Attribute>
<Attribute Name="Comment" AttributeDataType="xs:string">
<Value>Motor drive overheat</Value>
<Attribute Name="aml-lang=de-DE" AttributeDataType="xs:string">
<Value>Motor drive overheat</Value>

```

6.5 Hardware-Daten importieren/exportieren

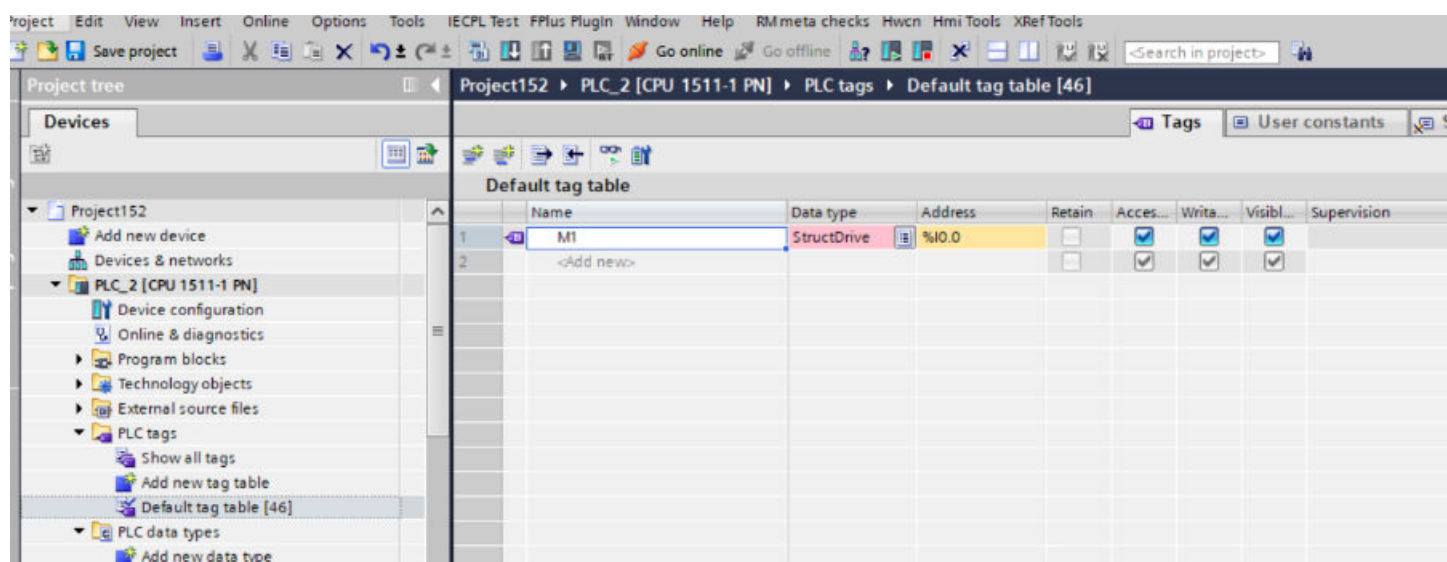
```

</Attribute>
</Attribute>
</ExternalInterface>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
ComplexTag" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
TagTable" />
</InternalElement>
<InternalElement ID="5983e7fd-ca6a-4573-8579-9ff323cfa09a" Name="PROFINET interface_1">
...
<InternalElement ID="2240a28c-ba88-4af4-ae6a-03d0ba3ce108" Name="E1">
...>
</InternalElement>
<InternalElement ID="df1d95d3-89ee-4a45-bca8-73e342497bee" Name="Port_1">
...
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="25f8dc3c-621d-4659-bb04-2d923df832c4" Name="Port_2">
...
</InternalElement>
<SupportedRoleClass
RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
Device" /></InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
AutomationProject" /></InternalElement>
</InstanceHierarchy>
</CAEXFile>

```

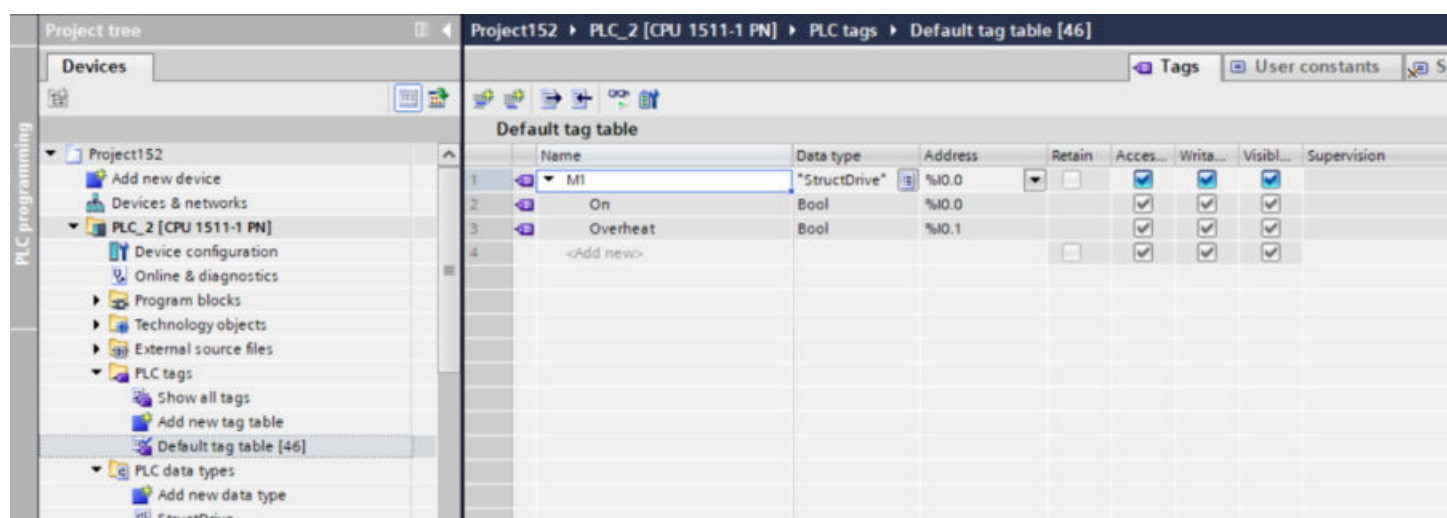
Import komplexer Variablen im TIA Portal ohne Erstellung von UDT

Obige Datei enthält eine komplexe Variable mit zwei Subvariablen 'On' und 'Overheat' Nach dem Importieren in TIA Portal wird eine komplexe Variable wie folgt erstellt:

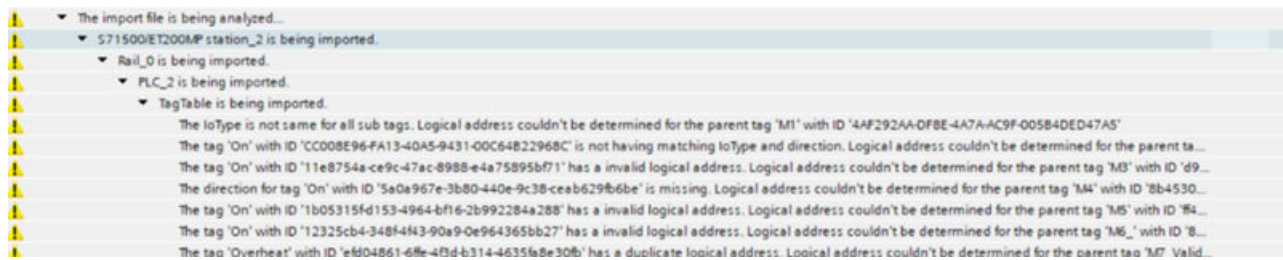


Import komplexer Variablen im TIA Portal mit Erstellung von UDT

Bei fehlendem oder ungültigem Datentyp wird für 'Data type' eine Fehlermeldung angezeigt. Im oben dargestellten Fall fehlt ein benutzerdefinierter Datentyp, nämlich StructDrive. Sobald der benutzerdefinierte Datentyp (StructDrive) angelegt ist, würden die komplexen Variablen wie folgt aussehen:



Bei Problemen mit einer der Subvariablen, zum Beispiel wenn deren logische Adresse fehlt, oder bei anderen Problemen in Bezug auf Richtung, IO-Typ usw., ist es nicht möglich, die niedrigste Adresse einer komplexen Variablen zu bestimmen. In diesem Fall werden komplexe Variablen ohne logische Adresse angelegt und dem Benutzer wird eine entsprechende Warnmeldung nach folgendem Beispiel angezeigt:



Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

6.5.14 Export von CAX-Daten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Einleitung

Im TIA Portal können Sie Ihre Konfiguration im Editor "Geräte & Netze" in eine AML-Datei exportieren. Mit dieser auf TIA Portal Openness basierenden Funktion lassen sich Hardwaredaten von der Projekt- oder Geräteebebene exportieren.

Mit der Export-Funktion von TIA Portal Openness können Sie CAX-Daten exportieren: Die Exportfunktion ist über den Dienst CaxProvider zugänglich. Über die Methode GetService am Objekt Project können Sie den Dienst CaxProvider aufrufen.

Mit der Benutzeroberfläche von TIA Portal V17 können CAX-Export- und -Importvorgänge in einer Multiuser-Umgebung durchgeführt werden. Wird ein Serverprojekt gleichzeitig mit der lokalen Sitzung geöffnet, finden Export und Import für das Serverprojekt statt. Um die lokale Sitzung zu exportieren oder zu importieren, muss das Serverprojekt geschlossen werden.

Für den Export eines Einzelgeräts als CAX muss ferner im Kontextmenü des Geräts der Export für ein Serverprojekt möglich sein, wenn es geöffnet ist. In diesem Fall darf das Kontextmenü für eine lokale Sitzung nicht angezeigt werden. Für den Export eines Einzelgeräts als CAX darf das Kontextmenü für eine lokale Sitzung nur verfügbar sein, wenn der Benutzer die Serverprojektsicht schließt.

In TIA Portal Openness V17 wird es auch möglich sein, einen CAX-Export und -Import in einer Multiuser-Umgebung auszuführen. Damit ein Multiuser-Projekt (lokale Sitzung und Serverprojekt) am Austausch über AML teilnehmen kann, wird eine neue API für den Export verwendet. Die neue Export-API nutzt 'ProjectBase' als Argument. Sie können jetzt mit der Export-API sowohl Single-User- als auch Multiuser-Projekte exportieren.

Ab TIA Portal Openness V17 wird von der Funktion für den CAX-Export/Import über die Benutzeroberfläche und die API Folgendes nicht mehr geprüft:

- Ob die Windows-Gruppe für Openness "Siemens TIA Openness" vorhanden ist
- Ob der Windows-Benutzer Mitglied der Gruppe "Siemens TIA Openness" ist

Hinweis

Wenn Sie CAX-APIs in einer Client-Applikation aufrufen möchten, müssen Sie sicherstellen, dass entweder die Applikation mit einem korrekten Openness-Zertifikat signiert ist oder dass die Openness-Gruppe "Siemens TIA Openness" vorhanden und der Benutzer Mitglied dieser Gruppe ist.

Bei TIA Portal V19 können Sie auf die Ergebnismeldungen von Cax-Export- und -Importvorgängen programmatisch zugreifen. Dies gibt Ihnen die Möglichkeit einer benutzerdefinierten Analyse und Visualisierung der Übertragungsergebnisse. Das Ergebnis enthält die Meldungen mit den benötigten Informationen.

Die neue Export-API für das Gerät und Projekt stellt als Rückgabewert ein strukturiertes TransferResult bereit und die Protokolldatei wird nicht generiert. Das Übertragungsergebnis können Sie verarbeiten und in einem benutzerdefinierten Format speichern.

Einschränkungen für CAX in Bezug auf das Exportieren und Importieren

CAX unterstützt das Exportieren und Importieren der folgenden Geräte nicht:

- HMI-Geräten außer Push Button Panels und Key Panels
- Antrieben

CAX unterstützt das Exportieren und Importieren bestimmter Scalance-Geräte in TIA Portal nicht.

- 6GK5 602-0BA10-2AA3
- 6GK5 612-0BA10-2AA3
- 6GK5 623-0BA10-2AA3
- 6GK5 627-2BA10-2AA3
- 6GK5 602-0BA00-2AA3
- 6GK5 612-0BA00-2AA3
- 6GK5 613-0BA00-2AA3
- System:Device.Scalance/S627
- 6GK5 495-8BA00-8AA2
- 6ES7 451-3AL00-0AE
- 6AV2 104-0xxxx-xxxx
- 6AV2 155-xxxxx-xxxx
- System:Rack.Scalance/X200BA-Y V4.2
- System:PC.DCSPlus.LogicalServer

- System:Device.DCS_ReducMCStation
- System:Device.DCS_MCStation
- System:Device.DCS_SimulationStation
- System:Device.OPCUAPackageUnit
- System:Device.IEC60870
- System:Device.S7Com
- System:Device.CS3000

Hinweis

Ab TIA Portal Openness V17 kann das normierte Format von Typenidentifizier für die oben angegebenen Geräteelemente auch ausgeschlossen werden.

Programmcode: Auf den Dienst CaxProvider zugreifen

Ändern Sie den folgenden Programmcode, um auf den Dienst CaxProvider zuzugreifen:

```
//For Single Project
ProjectBase project = tiaPortal.Projects.Open(...);
//Or
//for LocalSession of a Multiuser Project
MultiuserProject project = tiaPortal.LocalSessions.Open(...).Project;
//Or
//for Server Project of a Multiuser Project
MultiuserProject project = tiaPortal.LocalSessions.OpenServerProject(...).Project;
Or
//Single user Project
Project project = tiaPortal.Projects.Open(...);
CaxProvider caxProvider = project.GetService<CaxProvider>();
if(caxProvider != null)
{
// Perform CAX export and import operation
}
```

CAX-Export auf Projektebene

Um CAX-Daten auf Projektebene zu exportieren, können Sie ab TIA Portal V14 `Export()` mit folgenden Parametern verwenden:

Name	Typ	Beispiel	Beschreibung
ProjectToExport	Project	tiaPortal.Projects[0]	Projektobjekt zum Exportieren
ExportFilePath	FileInfo	new FileInfo(@"D:\Temp\ProjectExport.aml")	Vollständiger Exportdateipfad der AML-Datei
LogFilePath	FileInfo	new FileInfo(@"D:\Temp\ProjectExport_Log.log")	Vollständiger Dateipfad der Protokolldatei

Um CAX-Daten auf Projektebene zu exportieren, verwenden Sie ab TIA Portal V17 `Export()` mit folgenden Parametern:

Name	Typ	Beschreibung
ProjectToExport	ProjectBase	Zu exportierendes Single-User- oder Multiuser-Projekt
ExportFilePath	FileInfo	Vollständiger Exportdateipfad der AML-Datei
LogFilePath	FileInfo	Vollständiger Dateipfad der Protokolldatei

Rückgabetyt

Typ	Beschreibung
bool	"True", wenn die CAX-Export-Operation ohne Fehler ausgeführt wird, ansonsten "false"

Um CAX-Daten auf Projektebene zu exportieren, ändern Sie folgenden Programmcode:

```
caxProvider.Export(project, new FileInfo(@"D:\Temp\ProjectExport.aml"), new
FileInfo(@"D:\Temp\ProjectExport_Log.log"));
```

Überladungs-API für CAX-Export auf Projektebene

Um CAX-Daten auf Projektebene mit Zugriff auf die Ergebnismeldungen zu exportieren, können Sie ab TIA Portal V19 die Überladungs-Export-API mit folgenden Parametern verwenden:

Name	Typ	Beschreibung
projectToExport	ProjectBase	Zu exportierendes Single-User- oder Multiuser-Projekt
exportFilePath	FileInfo	Vollständiger Exportdateipfad der AML-Datei

Rückgabetyt

Typ	Beschreibung
TransferResult	Ergebnis einer CAX-Übertragung.

Die folgenden Eigenschaften werden bei `TransferResult` unterstützt:

Name der Eigenschaft	Typ	Zugriff
ErrorCount	int	Lesen
WarningCount	int	Lesen
State	TransferResultState	Lesen
Messages	TransferResultMessageComposition	Lesen

Die folgenden Eigenschaften werden bei `TransferResultMessageComposition` unterstützt:

Name der Eigenschaft	Typ	Zugriff
this[int]	TransferResultMessage	Lesen

Die folgenden Eigenschaften werden bei TransferResultMessage unterstützt:

Name der Eigenschaft	Datentyp	Zugriff
DateTime	DateTime	Lesen
ErrorCount	Int	Lesen
WarningCount	Int	Lesen
State	TransferResultState	Lesen
Message	String	Lesen

Liste der möglichen Ergebniszustände von Übertragungen:

Enum-Option	Beschreibung
TransferResultState.Success	Übertragung erfolgreich beendet
TransferResultState.Information	Übertragung mit Hinweisen beendet
TransferResultState.Warning	Übertragung mit Warnungen beendet
TransferResultState.Error	Übertragung mit Fehlern beendet

Um CAx-Daten auf Projektebene zu exportieren, ändern Sie folgenden Programmcode:

```
private static void CaxTransferAtProjectLevel(Siemens.Engineering.ProjectBase project,
CaxProvider caxProvider)
{
    FileInfo exportFilePath = new FileInfo("D:\\temp\\ExportFile.aml");
    // New Export API for project:
    TransferResult projectExportResult = caxProvider.Export(project, exportFilePath);
    PrintCaxResult(projectExportResult);
}
private static void PrintCaxResult(Siemens.Engineering.Cax.TransferResult result)
{
    Console.WriteLine($"CAx result summary: {result.State} (errors: {result.ErrorCount},
warnings: {result.WarningCount})");
    PrintCaxDetailResult(result.Messages);
}
private static void
PrintCaxDetailResult(Siemens.Engineering.Cax.TransferResultMessageComposition messages,
int nestingDepth = 0)
{
    foreach (Siemens.Engineering.Cax.TransferResultMessage message in messages)
    {
        string indent = new string(' ', nestingDepth * 2);
        Console.WriteLine($"{indent}{message.State} {message.Message} {message.DateTime} (errors:
{message.ErrorCount}, warnings: {message.WarningCount})");
        PrintCaxDetailResult(message.Messages, nestingDepth + 1);
    }
}
```

CAX-Export auf Geräteebe

Um CAX-Daten auf Geräteebe zu exportieren, verwenden Sie die Methode `Export` mit den folgenden Parametern:

Name	Typ	Beispiel	Beschreibung
DeviceToExport	Device	project.Devices[0]	Geräteobjekt zum Exportieren
ExportFilePath	FileInfo	new FileInfo(@"D:\Temp\ProjectExport.aml")	Vollständiger Exportdateipfad der AML-Datei
LogFilePath	FileInfo	new FileInfo(@"D:\Temp\ProjectExport_Log.log")	Vollständiger Dateipfad der Protokolldatei

Rückgabety

Typ	Beschreibung
bool	"True", wenn die CAX-Export-Operation ohne Fehler ausgeführt wird, ansonsten "false"

Um CAX-Daten auf Geräteebe zu exportieren, ändern Sie folgenden Programmcode:

```
caxProvider.Export(device, new FileInfo(@"D:\Temp\DeviceExport.aml"), new
FileInfo(@"D:\Temp\DeviceExport_Log.log"));
```

Überladungs-API für CAX-Export auf Geräteebe

Um CAX-Daten auf Geräteebe mit Zugriff auf die Ergebnismeldungen zu exportieren, können Sie ab TIA Portal V19 die Überladungs-Export-API mit folgenden Parametern verwenden:

Name	Typ	Beispiel	Beschreibung
deviceToExport	Device	project.Devices[0]	Zu exportierendes Geräteobjekt
exportFilePath	FileInfo	new FileInfo(@"D:\Temp\ProjectExport.aml")	Vollständiger Exportdateipfad der AML-Datei

Rückgabety

Typ	Beschreibung
TransferResult	Ergebnis einer CAX-Übertragung.

Die folgenden Eigenschaften werden bei `TransferResult` unterstützt:

Name der Eigenschaft	Typ	Zugriff
ErrorCount	int	Lesen
WarningCount	int	Lesen
State	TransferResultState	Lesen
Messages	TransferResultMessageComposition	Lesen

Die folgenden Eigenschaften werden bei TransferResultMessageComposition unterstützt:

Name der Eigenschaft	Typ	Zugriff
this[int]	TransferResultMessage	Lesen

Die folgenden Eigenschaften werden bei TransferResultMessage unterstützt:

Name der Eigenschaft	Datentyp	Zugriff
DateTime	DateTime	Lesen
ErrorCount	Int	Lesen
WarningCount	Int	Lesen
State	TransferResultState	Lesen
Message	String	Lesen

Liste der möglichen Ergebniszustände von Übertragungen:

Enum-Option	Beschreibung
TransferResultState.Success	Übertragung erfolgreich beendet
TransferResultState.Information	Übertragung mit Hinweisen beendet
TransferResultState.Warning	Übertragung mit Warnungen beendet
TransferResultState.Error	Übertragung mit Fehlern beendet

Um CAx-Daten auf Geräteebe zu exportieren, ändern Sie folgenden Programmcode:

```
private static void CaxTransferAtDeviceLevel(Siemens.Engineering.ProjectBase project,
CaxProvider caxProvider)
{
    FileInfo exportFilePath = new FileInfo("D:\\temp\\ExportFile.aml");
    Device deviceToExport = project.Devices.Find("Station_1");
    // New Export API for project:
    TransferResult deviceExportResult = caxProvider.Export(deviceToExport, exportFilePath);
    PrintCaxResult(deviceExportResult);
}
private static void PrintCaxResult(Siemens.Engineering.Cax.TransferResult result)
{
    Console.WriteLine($"CAx result summary: {result.State} (errors: {result.ErrorCount},
warnings: {result.WarningCount})");
    PrintCaxDetailResult(result.Messages);
}
private static void
PrintCaxDetailResult(Siemens.Engineering.Cax.TransferResultMessageComposition messages,
int nestingDepth = 0)
{
    foreach (Siemens.Engineering.Cax.TransferResultMessage message in messages)
    {
        string indent = new string(' ', nestingDepth * 2);
        Console.WriteLine($"{indent}{message.State} {message.Message} {message.DateTime} (errors:
{message.ErrorCount}, warnings: {message.WarningCount})");
        PrintCaxDetailResult(message.Messages, nestingDepth + 1);
    }
}
```

6.5.15 Import von CAx-Daten

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Verwendung

Im TIA Portal können Sie Ihre Konfiguration im Editor "Geräte & Netze" aus einer AML-Datei importieren. Mit dieser Funktion lassen sich Hardwaredaten von der Projekt- oder Geräteebe importieren.

Mit der Import-Funktion können Sie CAx-Daten importieren. Die Importfunktion ist über den Dienst `CaxProvider` zugänglich. Über die Methode `GetService` am Objekt `Project` können Sie den Dienst `CaxProvider` aufrufen.

Bei TIA Portal V19 stellt die neue Import-API als Rückgabewert ein strukturiertes `TransferResult` bereit und die Protokolldatei wird nicht generiert. Das Übertragungsergebnis können Sie verarbeiten und in einem benutzerdefinierten Format speichern.

Programmcode: Auf den Dienst `CaxProvider` zugreifen

Ändern Sie folgenden Programmcode:

```
//Access the CaxProvider service
Project project = tiaPortal.Projects.Open(...);
CaxProvider caxProvider = project.GetService<CaxProvider>();
if(caxProvider != null)
{
    // Perform Cax export and import operation
}
```

Cax-Import

Um CAX-Daten in ein TIA Portal-Projekt zu importieren, können Sie die Methode `Import` mit folgenden Parametern verwenden:

Name	Beispiel	Beschreibung
<code>ImportFilePath</code>	<code>new FileInfo(@"D:\Temp\ProjectExport.aml")</code>	Vollständiger Importdateipfad der AML-Datei
<code>LogFilePath</code>	<code>new FileInfo(@"D:\Temp\ProjectExport_Log.log")</code>	Vollständiger Dateipfad der Protokolldatei
<code>ImportOptions</code>	<code>CaxImportOptions.MoveToParkingLot</code> <code>CaxImportOptions.RetainTiaDevice</code> <code>CaxImportOptions.OverwriteTiaDevice</code>	Strategien zur Konfliktlösung beim Import in ein bereits vorhandenes, nicht leeres Projekt.

Um CAX-Daten zu importieren, ändern Sie folgenden Programmcode:

```
caxProvider.Import(new FileInfo(@"D:\Temp\ProjectImport.aml"), new
FileInfo(@"D:\Temp\ProjectImport_Log.log"), CaxImportOptions.MoveToParkingLot);
```

Die folgenden `CaxImportOptions` sind gegeben:

Importoption	Beschreibung
<code>MoveToParkingLot</code>	Gerät(e) mit Namenskonflikt im Projekt beibehalten und diejenigen aus den CAX-Daten in einen Parkplatzordner importieren
<code>RetainTiaDevice</code>	Gerät(e) mit Namenskonflikt im Projekt beibehalten und diejenigen aus den CAX-Daten nicht importieren
<code>OverwriteTiaDevice</code>	Gerät(e) mit Namenskonflikt im Projekt durch diejenigen aus den CAX-Daten überschreiben

Ab TIA Portal V18 wird beim CAX-Import eine Fehlermeldung angezeigt, wenn Produktlizenzen fehlen und versucht wird, eine Konfiguration, die eine Lizenz für die Erstellung erfordert, zu importieren. Eine Konfiguration, die keine Prüfung der Produktlizenz erfordert, wird ohne Fehler erstellt.

Beispiel: Der Import von Modulen wie S7-1500 PLCs (konfiguriert mit Typidentifizier, d. h. Artikelnummer in der AML-Datei) schlägt fehl, wenn die STEP 7-Produktlizenz fehlt, während der Import von Digitaleingabe-/ausgabebaugruppen (konfiguriert mit Typidentifizier, d. h. Artikelnummer in der AML-Datei) erfolgreich ist.

Auch eine Konfiguration, die auf 'Templatidentifizier' in der AML-Datei basiert, wird erfolgreich importiert, da eine bibliotheksbasierte Konfiguration keine Prüfung der Produktlizenz erfordert.

Beispiel: Der Import von Modulen wie S7-1500 PLCs, Digitalen Ein-Ausgabebaugruppen usw. (falls mit Templatidentifizier in der AML-Datei konfiguriert) wird auch bei fehlender STEP 7-Produktlizenz erfolgreich ausgeführt.

Eine fehlende Lizenz hat keinen Einfluss auf den CAX-Export. Der Export funktioniert daher wie zuvor.

Überladungs-API für CAX-Import

Um CAX-Daten auf Geräteebe mit Zugriff auf die Ergebnismeldungen in ein TIA Portal-Projekt zu importieren, können Sie ab TIA Portal V19 die Überladungs-Import-API mit folgenden Parametern verwenden:

Name	Beispiel	Beschreibung
ImportFilePath	new FileInfo(@"D:\Temp\ProjectExport.aml")	Vollständiger Importdateipfad der AML-Datei
ImportOptions	CaxImportOptions.MoveToParkingLot CaxImportOptions.RetainTiaDevice CaxImportOptions.OverwriteTiaDevice	Strategien zur Konfliktlösung beim Import in ein bereits vorhandenes, nicht leeres Projekt.

Rückgabetypp

Typ	Beschreibung
TransferResult	Ergebnis einer CAX-Übertragung.

Die folgenden Eigenschaften werden beim Typ TransferResult unterstützt:

Name der Eigenschaft	Typ	Zugriff
ErrorCount	int	Lesen
WarningCount	int	Lesen
State	TransferResultState	Lesen
Messages	TransferResultMessageComposition	Lesen

Die folgenden Eigenschaften werden bei TransferResultMessageComposition unterstützt:

Name der Eigenschaft	Typ	Zugriff
this[int]	TransferResultMessage	Lesen

Die folgenden Eigenschaften werden bei TransferResultMessage unterstützt:

Name der Eigenschaft	Datentyp	Zugriff
DateTime	DateTime	Lesen
ErrorCount	Int	Lesen
WarningCount	Int	Lesen
State	TransferResultState	Lesen
Message	String	Lesen

Liste der möglichen Ergebniszustände von Übertragungen:

Enum-Option	Beschreibung
TransferResultState.Success	Übertragung erfolgreich beendet
TransferResultState.Information	Übertragung mit Hinweisen beendet
TransferResultState.Warning	Übertragung mit Warnungen beendet
TransferResultState.Error	Übertragung mit Fehlern beendet

Um CAX-Daten zu importieren, ändern Sie folgenden Programmcode:

```
private static void ImportCaxTransfer(Siemens.Engineering.ProjectBase project, CaxProvider
caxProvider)
{
    FileInfo importFilePath = new FileInfo("D:\\temp\\ImportFile.aml");
    CaxImportOptions importOption = Siemens.Engineering.Cax.CaxImportOptions.RetainTiaDevice;
    // New Import API:
    TransferResult importResult = caxProvider.Import(importFilePath, importOption);
    PrintCaxResult(importResult);
}
private static void PrintCaxResult(Siemens.Engineering.Cax.TransferResult result)
{
    Console.WriteLine($"CAX result summary: {result.State} (errors: {result.ErrorCount},
warnings: {result.WarningCount})");
    PrintCaxDetailResult(result.Messages);
}
private static void
PrintCaxDetailResult(Siemens.Engineering.Cax.TransferResultMessageComposition messages,
int nestingDepth = 0)
{
    foreach (Siemens.Engineering.Cax.TransferResultMessage message in messages)
    {
        string indent = new string(' ', nestingDepth * 2);
        Console.WriteLine($"{indent}{message.State} {message.Message} {message.DateTime} (errors:
{message.ErrorCount}, warnings: {message.WarningCount})");
        PrintCaxDetailResult(message.Messages, nestingDepth + 1);
    }
}
```


6.5.16 Export/Import von Submodulen

Voraussetzungen

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Der PLC ist offline.

Anwendung

Zwischen dem TIA Portal und anderen Engineering-Tools, z. B. einem CAD-Tool wie EPLAN, können Submoduldaten im Round-Trip-Verfahren ausgetauscht werden, wobei während des Exports und Imports eine gemeinsame Hierarchie der Submodule in der AML-Datei beibehalten werden muss. Beispielsweise müssen die Submodule, wie etwa Busadapter, in TIA Portal eine andere interne Hierarchie aufweisen als in anderen Anwendungen (z. B. CAD-Tools wie EPLAN).

AML-Struktur der Exportdatei

Sie können Submoduldaten aus der TIA Portal-Hierarchie in die AML-Dateihierarchie exportieren.

Das folgende Beispiel zeigt einen Ausschnitt der AML-Dateistruktur, die beim Exportieren eines Busadapters als Submodul aus dem TIA Portal erzeugt wird.

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="Project4.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
<AdditionalInformation>
<WriterHeader>
<WriterName>Totally Integrated Automation Portal</WriterName>
<WriterID>1d4fceb-1ad6-4881-b01d-bca335d94a46:V1.0</WriterID>
<WriterVendor>Siemens AG</WriterVendor>
<WriterVendorURL>www.siemens.com</WriterVendorURL>
<WriterVersion>15</WriterVersion>
<WriterRelease>1500.0100.0.0</WriterRelease>
<LastWritingDateTime>2018-05-03T11:23:10.3011329Z</LastWritingDateTime>
</WriterHeader>
</AdditionalInformation>
<AdditionalInformation AutomationMLVersion="2.0" />
<AdditionalInformation DocumentVersions="Recommendations">
<Document DocumentIdentifier="AR APC" Version="1.1.0" />
</AdditionalInformation>
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
<InternalElement ID="6cd7f80f-e049-4958-ba67-630481805bf0" Name="Project4">
<Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
<Attribute Name="ProjectSign" AttributeDataType="xs:string" />
<Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
<Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
<InternalElement ID="b27045c4-9cb3-4b8d-916b-85f8100d1602" Name="Ungrouped devices">
<InternalElement ID="3f770698-940d-49c2-9f77-06fc458e1340" Name="ET 200SP station_1">
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>System:Device.ET200SP</Value>
</Attribute>
<InternalElement ID="6f52fbab-a221-4d54-9368-84c392ca7fec" Name="Rack_0">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>Rack</Value>
...
<InternalElement ID="f7445c0b-1c52-4a84-915f-2c8bee13af70" Name="BA 2xRJ45">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>BA 2xRJ45</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>127</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 193-6AR00-0AA0</Value>
</Attribute>
<Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
<Value>V0.0</Value>
</Attribute>
<InternalElement ID="40f8bbce-35d3-4d65-907a-bece3e0144e0" Name="PROFINET interface">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">

```

6.5 Hardware-Daten importieren/exportieren

```

<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<InternalElement ID="8fb775eb-96c6-48d6-af8a-96ba72418830" Name="IE1">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Ethernet</Value>
</Attribute>
<Attribute Name="NetworkAddress" AttributeDataType="xs:string">
<Value>192.168.0.1</Value>
</Attribute>
<Attribute Name="SubnetMask" AttributeDataType="xs:string">
<Value>255.255.255.0</Value>
</Attribute>
<Attribute Name="IpProtocolSelection" AttributeDataType="xs:string">
<Value>Project</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
</InternalElement>
<InternalElement ID="f28a3d93-d821-4556-9df1-a45f0e4ff6a6" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P1R</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="ad6a0faa-3b70-4528-8c54-8183018b6714" Name="Port_2">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P2R</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
...

```

Hinweis

CAX exportiert und importiert die Version von AR APC in eine AML-Datei gemäß der installierten Version des TIA Portals.

Import von Submodulen

Sie können Submodule aus AML-Dateien importieren, die beim vorhergehenden Export erzeugt wurden.

Hinweis

- Das veränderte Verhalten der Exporthierarchie gilt erst ab Version V15.1 aufwärts.
 - Die Hierarchie in der AML-Datei darf die interne Hierarchie in TIA Portal nach dem Import nicht beeinflussen.
 - Mit älteren TIA Portal-Versionen erzeugte AML-Dateien sollten ebenfalls fehlerfrei importiert werden.
 - Dieses Verhalten in Bezug auf die Änderung/Umwandlung der Hierarchie gilt sowohl für eingebaute als auch für nicht eingebaute Submodule.
-

Mehrere Submodule an einer gemeinsamen Schnittstelle

In manchen Fällen sind mehrere Submodule an einer gemeinsamen Schnittstelle vorhanden. Beispiel: IO-Device: IM 155-6 PN/3 HF 6ES7 155-6AU30-0CN0/V4.2. Dieses Modul verfügt über zwei nicht eingebaute Busadapter an der gleichen Schnittstelle. In diesem Fall muss es möglich sein, die Busadapter aus der TIA Portal-Hierarchie in die erforderliche AML-Dateihierarchie zu exportieren. In diesem Beispiel aus der TIA Portal-Hierarchie hat die PROFINET-Schnittstelle zwei Busadapter, drei Ports und einen Knoten. Hier gehören Port_1 und Port_2 logisch zu BA 2xRJ45 und Port_3 gehört logisch zu BA 2xRJ45_1, obwohl alle drei Ports an einer Schnittstelle zusammengeführt sind.

Beim Export:

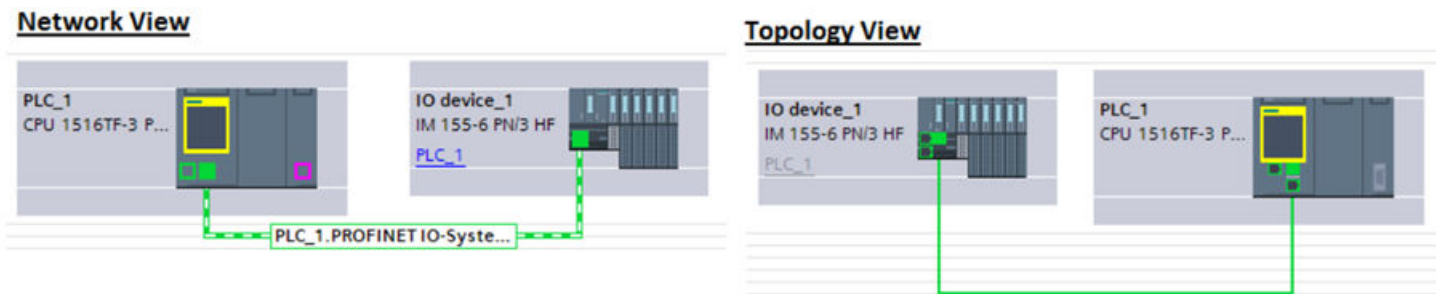
- Die AML-Datei muss mit den Attributen Label und Type zur Unterstützung der Sekundärschnittstelle exportiert werden
- Nur dem ersten Submodul wird die Original-Schnittstelle mit den verbindungsrelevanten Informationen zugewiesen. Hier wird BA 2xRJ45 die Original-Schnittstelle mit Knoten 'IE1', 'Port_1' und 'Port_2' zugewiesen.
- Die anderen Submodule werden einer 'duplizierten' Schnittstelle mit Ports zugewiesen, die logisch zum Submodul gehören. Hier wird BA 2xRJ45_1 eine 'duplizierte' Schnittstelle und Port_3 zugewiesen.
- Wenn das Kopfmodul mit einem Subnetz/IO-System verbunden ist, wird die zugehörige Verbindungsinformation (wie ExternalInterface-Links) nur als Teil des ersten Submoduls exportiert (ExternalInterface-Link zum Subnetz unter 'Node' und ExternalInterface-Link zum IO-System unter 'Interface').
- Die Verbindungsinformationen zur Topologieverschaltung gehören zum jeweiligen Port.

Beim Import:

- Es ist möglich, mehrere Submodule aus einer AML-Datei zu importieren, die beim vorhergehenden Export erzeugt wurde.
- Es ist möglich, dass eine von EPLAN erzeugte AML-Datei in der sekundären Schnittstelle Knoteninformationen enthält.

- Knotendetails, die ein Duplikat aus dem Knoten der primären Schnittstelle sind, werden wie folgt gehandhabt:
 - Knotenattribute: Details zu Knotenattributen, die während der Verarbeitung der primären Schnittstelle festgelegt wurden, werden überschrieben.
 - Subnetzverbindung: Wird stillschweigend übersprungen, wenn die Verbindung bereits besteht, andernfalls wird die Verbindung hergestellt.
- Wenn die AML-Datei Details zur IO-Systemverbindung an der sekundären Schnittstelle enthält, dann
 - Wird die Verbindungsherstellung übersprungen, sofern die Verbindung bereits vorhanden ist, und der Anwender wird über eine entsprechende Fehlermeldung in der Registerkarte "Info" benachrichtigt.
 - Wird die Verbindung hergestellt, sofern sie noch nicht vorhanden ist.

Die folgende Konfiguration zeigt eine IO-Device-Konfiguration mit Master-Slave-Verbindungen und Topologieverbindungen.



Das folgende Beispiel zeigt einen Ausschnitt einer AML-Datei, die beim Export für obige Konfiguration erzeugt wird:

6.5 Hardware-Daten importieren/exportieren

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="MultipleBA_01.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
<AdditionalInformation>
<WriterHeader>
<WriterName>Totally Integrated Automation Portal</WriterName>
<WriterID>1d4fceb-1ad6-4881-b01d-bca335d94a46:V1.0</WriterID>
<WriterVendor>Siemens AG</WriterVendor>
<WriterVendorURL>www.siemens.com</WriterVendorURL>
<WriterVersion>15</WriterVersion>
<WriterRelease>1501.0000.0.0</WriterRelease>
<LastWritingDateTime>2018-05-17T09:36:46.9230179Z</LastWritingDateTime>
</WriterHeader>
</AdditionalInformation>
<AdditionalInformation AutomationMLVersion="2.0" />
<AdditionalInformation DocumentVersions="Recommendations">
<Document DocumentIdentifier="AR APC" Version="1.1.0" />
</AdditionalInformation>
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
<InternalElement ID="e005c094-1b0a-42c4-92a0-67c981508cla" Name="Project45">
<Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
<Attribute Name="ProjectSign" AttributeDataType="xs:string" />
<Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
<Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
<InternalElement ID="2782e61d-8c27-46cb-93ea-6b804157ae60" Name="PN/IE_1">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Ethernet</Value>
</Attribute>
<ExternalInterface ID="2d901881-a2bf-4fe7-915f-b2542b346988" Name="LogicalEndPoint_Subnet"
RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
Subnet" />
...
<InternalElement ID="dc5cf410-2516-4b0b-ad1a-c43117d8c9b3" Name="BA 2xRJ45">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>BA 2xRJ45</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>127</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 193-6AR00-0AA0</Value>
</Attribute>
<Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
<Value>V0.0</Value>
</Attribute>
<InternalElement ID="f04874a8-2d35-47c4-93ae-d6fdc2668479" Name="PROFINET interface">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>

```



```

</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="81a7d9df-99b8-4eca-8e72-404b22bd05e7"
Name="LogicalEndPoint_Interface" RefBaseClassPath="CommunicationInterfaceClassLib/
LogicalEndPoint" />
<InternalElement ID="83eb7d69-8cd5-4217-a07a-0c656d215ec7" Name="IE1">
...

```

Geräte mit Submodul und integriertem Port auf Schnittstellenebene

Es gibt bestimmte Gerätekonfigurationen, bei denen ein Modul einen integrierten Port sowie erweiterte Ports unterstützen kann (z. B. über ein Submodul, d. h. Busadapter). Beispiel: ET200SP CPU: CPU 1510SP-1 PN/6ES7 510-1DJ00-0AB0/V1.8. Dieses CPU-Modul unterstützt drei Ports (zwei Ports über den Busadapter und einen integrierten Port) an derselben Schnittstelle.

In diesem Beispiel aus der TIA Portal-Hierarchie hat die PROFINET-Schnittstelle einen Busadapter, drei Ports und einen Knoten. Hier gehören Port_1 und Port_2 logisch zu BA 2xRJ45 und Port_3 gehört logisch zur PROFINET-Schnittstelle, obwohl alle drei Ports an einer Schnittstelle zusammengeführt sind.

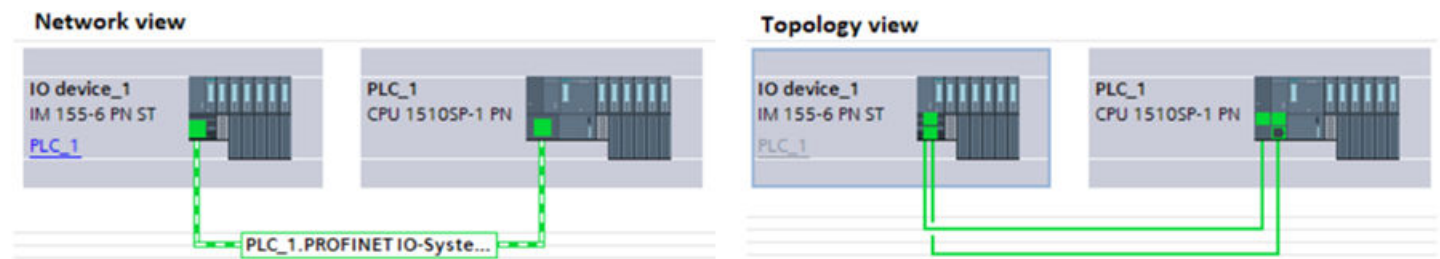
Beim Export:

- Dem Submodul wird die Original-Schnittstelle mit den verbindungsrelevanten Informationen zugewiesen. Hier wird BA 2xRJ45 die Original-Schnittstelle mit 'IE1', 'Port_1' und 'Port_2' zugewiesen. Die 'duplizierte' Schnittstelle (Typ 'Ethernet') des Kopfmoduls hat nur den integrierten Port 'Port_3'.
- Für den Fall, dass das Submodul (BA 2xRJ45) nicht gesteckt ist, ist die PROFINET-Schnittstelle auf Kopfmodulebene als Original-Schnittstelle mit 'IE1' und 'Port_3' zu betrachten.
- Wenn das Kopfmodul mit einem Subnetz/IO-System verbunden ist, wird die zugehörige Verbindungsinformation (wie ExternalInterface-Links) nur als Teil der Original-Schnittstelle exportiert (ExternalInterface-Link zum Subnetz unter 'Node' und ExternalInterface-Link zum IO-System unter 'Interface').
- Die Verbindungsinformationen zur Topologieverschaltung gehören zum jeweiligen Port.

Beim Import:

- Es ist möglich, Module mit einer Schnittstelle mit einem Submodul und integrierten Port aus einer AML-Datei zu importieren, die beim vorhergehenden Export erzeugt wurde.
- Die Handhabung redundanter Informationen (Knoten, IO-System und Links) an der 'duplizierten' Schnittstelle hat genauso zu erfolgen wie bei dem Szenario 'Mehrere Submodule an einer gemeinsamen Schnittstelle'.

Die folgende Konfiguration zeigt Module mit Master-Slave-Verbindungen und Topologieverbindungen.



Die AML-Datei, die beim Export für obige Konfiguration erzeugt wird, ist nachfolgend dargestellt:

6.5 Hardware-Daten importieren/exportieren

```

<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="Project_BusAdapter_Demo.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
...
<AdditionalInformation AutomationMLVersion="2.0" />
<AdditionalInformation DocumentVersions="Recommendations">
<Document DocumentIdentifier="AR APC" Version="1.2.0" />
</AdditionalInformation>
<InternalElement ID="12b43940-cea6-476a-886c-11ebaa518256" Name="BA 2xRJ45">
...
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 193-6AR00-0AA0</Value>
</Attribute>
<Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
<Value>V0.0</Value>
</Attribute>
<InternalElement ID="8f8a4cdb-8f4b-4e72-8c92-b8falcc3bf70" Name="PROFINET interface_1">
...
<InternalElement ID="7a9938c3-ccc0-4d28-be35-333a343f3613" Name="E1">
...
<ExternalInterface ID="3d9f7f55-723f-4c5d-a2e7-ffe1ec3b9167" Name="LogicalEndPoint_Node"
RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationEthernetRoleClassLib/
NodeEthernet" /></InternalElement>
<InternalElement ID="3c4085cb-7565-4673-8de1-c5624c4c08dc" Name="PROFINET IO-System">
<Attribute Name="Number" AttributeDataType="xs:int">
<Value>100</Value>
</Attribute>
<ExternalInterface ID="7fb6129f-95c4-4d2c-aab5-702937198e80"
Name="LogicalEndPoint_IoSystem" RefBaseClassPath="CommunicationInterfaceClassLib/
LogicalEndPoint" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
IoSystem" />
</InternalElement>
<InternalElement ID="1f1d3e8d-55da-4355-b87e-7feb58d86143" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P1R</Value>
</Attribute>
...
<ExternalInterface ID="850e3f32-985f-4432-b627-26e2775a69cc"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="11731ed5-7922-41c9-b179-a5ae029cc10d" Name="Port_2">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P2R</Value>
</Attribute>
...
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" /> </InternalElement>

```

```

<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<InternalElement ID="7be36254-5ed6-4a6f-9e7b-90be8b35e595" Name="PROFINET interface_1">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Ethernet</Value>
</Attribute>
...
<InternalElement ID="61ac6187-8a5d-4f98-ae91-b809c0a3a15d" Name="Port_3">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P3</Value>
</Attribute>
...
<ExternalInterface ID="425f5a5d-84e2-40c6-928f-e1aab73a8b86"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>

```

Pruning von AML-Dateien

Das Pruning (Bereinigen) beschreibt den Vorgang der Optimierung des Inhalts durch Entfernen bestimmter Dinge, die nicht unbedingt angegeben werden müssen. Informationen über das Pruning von AML-Dateien enthält Pruning von AML-Dateien (Seite 1478).

In manchen Fällen kann die Hierarchie der Submodule in TIA Portal und CAD-Tools (wie EPLAN) auf Grund von Pruning der Submodule voneinander abweichen. In diesen Fällen muss TIA Portal den Import von AML-Dateien sowohl mit als auch ohne Pruning unterstützen.

Hinweis

- Der Export von AML-Dateien ohne Pruning wird in TIA Portal immer unterstützt.
 - Der Import von AML-Dateien, ob mit oder ohne Pruning, wird in TIA Portal immer unterstützt.
-

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.17 AML-Dateien in einer UMAC-Umgebung exportieren/importieren

Einleitung

Ab TIA Portal Openness ab V17 ist es möglich, einen CAX-Export und -Import für ein UMAC-geschütztes Projekt durchzuführen.

CAX-Vorgänge für ein geschütztes Projekt beruhen auf den folgenden Funktionsrechten:

- Projekt mit Lese- und Schreibzugriff
- Projekt über Openness API ändern

Export

Der CAX-Export ist nicht beschränkt. Daher können Export-Vorgänge für geschützte Projekte unabhängig von den oben erwähnten Funktionsrechten der Benutzer durchgeführt werden.

Import

Der CAX-Import ist beschränkt. Wenn ein Benutzer obige Zugriffsrechte besitzt, kann der Import erfolgreich ausgeführt werden, sonst wird eine Fehlermeldung wegen fehlender Funktionsrechte in der Benutzeroberfläche von TIA Portal angezeigt oder eine Ausnahme wegen nicht gefundener Funktionsrechte wird für die CAX API ausgelöst.

Funktionsrechte von Benutzern und CAX-Vorgänge

Die folgende Tabelle zeigt die Funktionsrechte der Benutzer und die zulässigen CAX-Vorgänge:

Benutzer	Funktionsrechte		CAX-Vorgänge	
	Projekt mit Lese- und Schreibzugriff	Projekt über Openness API ändern	Export	Import
Projektadministrator	X	X	X	X
Benutzer nur mit Lesezugriff auf das Projekt	-	X	X	-
Benutzer ohne Openness-Zugriff	X	-	X	-
Benutzer nur mit Lesezugriff und ohne Openness-Zugriff	-	-	X	-

Hinweis

- Ab TIA Portal V17 kann ein CAX-Export auch für schreibgeschützte Projekte ausgeführt werden.

Für das Exportieren und Importieren eines UMAC-geschützten Projekts mit der CAX API muss das Projekt in einer UMAC-geschützten Umgebung geöffnet werden. Zum Öffnen eines Projekts in einer UMAC-geschützten Umgebung, siehe Öffnen eines Projekts (Seite 131) über API.

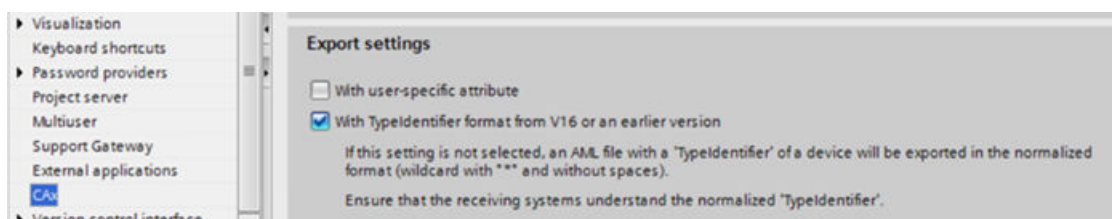
Siehe auch

Projekt öffnen (Seite 131)

6.5.18 AML-Datei mit normiertem Typbezeichner exportieren/importieren**Einleitung**

In TIA Portal gibt es verschiedene Darstellungen der Bestellnummern innerhalb der Typbezeichner. Damit diese verschiedenen Darstellungen der Bestellnummern in andere Tools importiert und exportiert werden können, ist jetzt eine tolerante Handhabung der Bestellnummern in AML-Dateien möglich.

Für den Export des Typidentifier-Formats aus TIA Portal V16 und niedriger wurde eine neue Einstellung hinzugefügt:



Wird dieses Kontrollkästchen "Mit Typidentifier-Format aus V16 und niedriger" aktiviert, wird der Typidentifier-Wert in AML im alten Format exportiert, z. B. OrderNumber:6ES7 516-3AN00-0AB0.

Wird das Kontrollkästchen nicht aktiviert, wird der Typidentifier-Wert der AML-Datei mit Wildcard-Zeichen ("*") für die Bestellnummer exportiert und Leerzeichen werden entfernt; Beispiel: OrderNumber:6ES7590-1***0-0AA0.

Während des Imports werden willkürliche Wildcard-Zeichen und Leerzeichen als Teil der Bestellnummer in einem Typidentifier akzeptiert.

6.5.19 CAX-Daten ohne logische Adresse importieren**Voraussetzung**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Beim CAX-Import in TIA Portal können Sie die Verbindung zwischen Kanal und Variable konfigurieren, ohne dass Sie die Anfangsadresse eines E/A-Moduls und/oder die logische Adresse der Variablen in der AML-Datei angegeben haben.

Das folgende AML-Beispiel zeigt die XML-Datei, die ohne Attribute für Anfangsadresse und logische Adresse erzeugt werden soll.

6.5 Hardware-Daten importieren/exportieren

```

<?xml version="1.0" encoding="utf-8"?><CAEXFile FileName="TagsExport.aml"
SchemaVersion="2.15" xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
...
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
<InternalElement ID="fff25423-fe9e-4334-9331-4cec118e06f7" Name="Project1">
...
<InternalElement ID="59c0b48b-aa6c-45c3-8dc8-bba5367bd4fb" Name="S7300/ET200M station_1">
...
<InternalElement ID="1b7b2b24-243b-4348-831e-bc46bc35957f" Name="Rail_0">
...
<InternalElement ID="974ca791-ad8d-482b-be80-2cf4e8dcedaf" Name="PLC_1">
...
<InternalElement ID="9564bcc2-8ea0-4be7-a950-5c55b34e474a" Name="Default tag table">
<ExternalInterface ID="7fd969e6-c2c9-45a8-b573-68833df327f5" Name="Tag_1"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Tag">
<Attribute Name="DataType" AttributeDataType="xs:string">
<Value>Bool</Value>
</Attribute>
</ExternalInterface>
<ExternalInterface ID="33899862-86c1-4171-832a-1136b6e59b9d" Name="Tag_2"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Tag">
<Attribute Name="DataType" AttributeDataType="xs:string">
<Value>Byte</Value>
</Attribute>
</ExternalInterface>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
TagTable" />
</InternalElement>
...
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>8</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<Attribute Name="Address">
<RefSemantic CorrespondingAttributePath="OrderedListType" />
<Attribute Name="1">
<Attribute Name="StartAddress" AttributeDataType="xs:int">
<Value>832</Value>
</Attribute>
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>128</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
</Attribute>
<Attribute Name="2">
<Attribute Name="StartAddress" AttributeDataType="xs:int">
<Value>832</Value>
</Attribute>
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>128</Value>
</Attribute>

```

```

<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Output</Value>
</Attribute>
</Attribute>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<InternalElement ID="29e0bb63-0050-46e3-968a-fcecf4eb050a" Name="DI 16x24VDC_1">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>DI16 x 24VDC</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>4</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 321-1BH02-0AA0</Value>
</Attribute>
<Attribute Name="Address">
<RefSemantic CorrespondingAttributePath="OrderedListType" />
<Attribute Name="1">
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>16</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
</Attribute>
</Attribute>
<ExternalInterface ID="175dc9c9-f9a3-4b10-b43e-68dfc14811fc" Name="Channel_DI_0"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Digital</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
<Attribute Name="Number" AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
</ExternalInterface>
<ExternalInterface ID="23e99053-906c-4548-9bd2-e975cacf01b2" Name="Channel_DI_1"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Digital</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">

```

6.5 Hardware-Daten importieren/exportieren

```

<Value>Input</Value>
</Attribute>
<Attribute Name="Number" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
</ExternalInterface>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
<InternalLink Name="Link To Tag_1" RefPartnerSideA="29e0bb63-0050-46e3-968a-
fcec4eb050a:Channel_DI_0" RefPartnerSideB="9564bcc2-8ea0-4be7-a950-5c55b34e474a:Tag_1" />
<InternalLink Name="Link To Tag_2" RefPartnerSideA="29e0bb63-0050-46e3-968a-
fcec4eb050a:Channel_DI_0" RefPartnerSideB="9564bcc2-8ea0-4be7-a950-5c55b34e474a:Tag_2" />
<InternalLink Name="Link To Tag_3" RefPartnerSideA="29e0bb63-0050-46e3-968a-
fcec4eb050a:Channel_DI_1" RefPartnerSideB="9564bcc2-8ea0-4be7-a950-5c55b34e474a:Tag_2" />
<InternalLink Name="Link To Tag_4" RefPartnerSideA="29e0bb63-0050-46e3-968a-
fcec4eb050a:Channel_DI_2" RefPartnerSideB="9564bcc2-8ea0-4be7-a950-5c55b34e474a:Tag_2" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
Device" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
AutomationProject" />
</InternalElement>
</InstanceHierarchy>
</CAEXFile>

```

Variable vom Datentyp Bool (%I0.0)

Im obigen Beispiel einer AML-Datei wird beim Import die logische Adresse für die Variable mit Hilfe des folgenden Algorithmus berechnet:

Logische Adresse = ChannelloType + ByteAddress + BitAddress

Name	Beschreibung
ChannelloType	Eingang (E) oder Ausgang (A)
ByteAddress	ByteAddress wird wie folgt berechnet: StartAddress des E/A-Moduls * 8 + BitOffsetAddress des E/A-Moduls + (Channel-Number * ChannelLength) / 8
BitAddress	BitAddress wird wie folgt berechnet: (BitOffsetAddress des E/A-Moduls + (ChannelNumber * ChannelLength)) % 8

Hinweis

- Wenn eine Variable modulübergreifend ist: Der Algorithmus oben liefert mehrere Byte-Adressen basierend auf der Anfangsadresse der Module sowie ein Array aus Bitadressen pro Byte-Adresse entsprechend den einzelnen Kanalnummern. Dann wählt der Algorithmus die niedrigste Byte-Adresse und die niedrigste Bitadresse im Array entsprechend diesem Byte aus, um die logische Adresse der Variablen zu berechnen. Sobald die logische Adresse der Variablen zugeordnet ist, wird die übergreifende Eigenschaft beim Import automatisch vom TIA Portal berücksichtigt.
- Im TIA Portal unterstützen nur wenige Gerätekonfigurationen (zum Beispiel ASI-Module) das BitOffset-Adressattribut. Für Module, die das BitOffset-Adressattribut nicht unterstützen, wird für obige Berechnung der Standardwert "0" angenommen.

Die folgende Liste enthält die Variablen-Datentypen, die die Bit-Adresse im TIA Portal unterstützen:

Variablen-Datentypen	Wert der Bitadresse
Bool	0 bis 7
LReal	Die höchsten und niedrigsten Werte für Bitnummern im TIA Portal sind 0.
LWord	
LInt	
ULInt	
LTime	
LDT	
LTime_Of_Day	

Wenn beim CAx-Import für einen Kanaltyp eine Boolesche Variable konfiguriert ist, umfasst die Berechnung der logischen Adresse die Bit-Offset-Adresse. Die logische Adresse zusammen mit dem Bit-Offset wird als logische Adresse der Variablen in der Oberfläche des TIA Portals aktualisiert.

Variablen mit anderen Datentypen, die die BitOffset-Adresse unterstützen

Wenn zwischen zwei verschiedenen Datentypen ein Variablenkanal konfiguriert ist, wobei ein Kanal des Typs Bool einer mehrere Kanäle überspannenden Variablen des Typs LDT zugeordnet ist, umfasst die Berechnung der logischen Adresse die Bit-Offset-Adresse für die Berechnung der logischen Adresse der Variablen des Datentyps "LDT". Die logische Adresse der Variablen wird in der Oberfläche im TIA Portal mit einem Fehler aktualisiert, wenn der Wert der Bitadresse ein anderer als "0" ist; die Kanalkonfiguration der Variablen ist dann nicht möglich.

Es muss sichergestellt werden, dass die Variablen-Kanal-Konfiguration zwischen gleichen Datentypen ausgeführt wird.

Variable mit anderem Datentyp (%IB0)

Logische Adresse = ChannelloType + TagDataType + ByteAddress

Name	Beschreibung
ChannelloType	Eingang (E) oder Ausgang (A)
TagDataType	TagDataType ist eine Abkürzung des Variablentyps. Beispiel: W für Wort und B für Byte
ByteAddress	ByteAddress wird wie folgt berechnet: $\text{StartAddress des E/A-Moduls} + (\text{ChannelNumber} * \text{ChannelLength}) / 8$

Der oben beschriebene Algorithmus wird für die genaue Berechnung der logischen Adresse einer Variablen in folgenden Fällen verwendet:

- Die Länge des in der Variablen angegebenen Datentyps sollte gleich der Länge des Kanals sein, dem er zugewiesen ist.
- Wird beispielsweise eine Variable vom Datentyp "Byte" einem analogen Kanal mit einer Länge von 2 Bytes zugewiesen, muss beim Import einer AML-Datei ohne logische Adresse der Variablen die Variable im TIA Portal immer dem ersten Byte des Kanals zugewiesen werden, unabhängig davon, welchem Byte sie ursprünglich zugewiesen war.

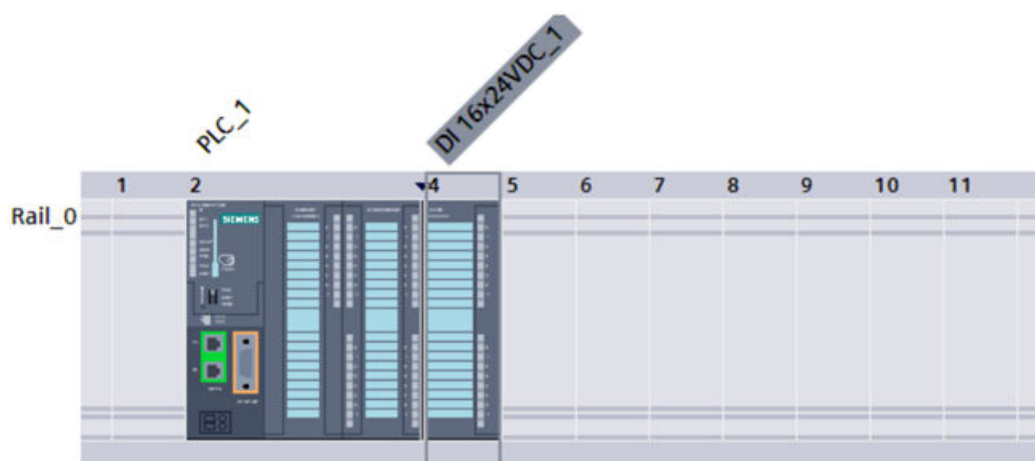
Hinweis

- Wenn eine Variable modulübergreifend ist: Der Algorithmus oben liefert mehrere Byte-Adressen basierend auf der Anfangsadresse der Module. Dann wählt der Algorithmus die niedrigste Byte-Adresse aus, um die logische Adresse der Variablen zu berechnen. Sobald die logische Adresse mit der niedrigsten Byte-Adresse der Variablen zugeordnet ist, wird die übergreifende Eigenschaft beim Import automatisch vom Portal gehandhabt. Wenn das Attribut StartAddress für das E/A-Modul in der AML-Datei nicht angegeben ist, wird vom TIA Portal der Standardwert zugeordnet, der dann für obige Berechnung verwendet wird.
- Wenn das Attribut StartAddress für das E/A-Modul in der AML-Datei nicht angegeben ist, wird vom TIA Portal der Standardwert zugeordnet, der dann für obige Berechnung verwendet wird.

Nach erfolgreichem Import wurde im TIA Portal die folgende Variablenkonfiguration für das oben beschriebene Beispiel erstellt.

Default tag table						
	Name	Data type	Address	Retain	Acces...	Visibl...
1	Tag_1	Bool	%I0.0		☒	☒
2	Tag_2	Byte	%IB0		☒	☒
3	<Add new>				☒	☒

Die Kanäle sind mit den Variablen wie unten dargestellt zu konfigurieren.



Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

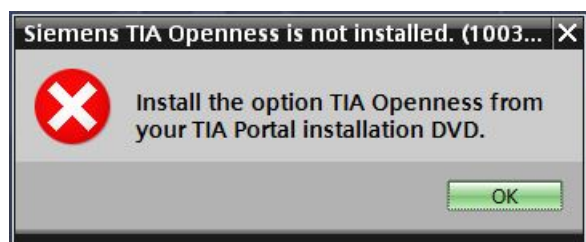
Projekt öffnen (Seite 131)

6.5.20 Ausnahmen beim Import und Export von CAX-Daten

Ausnahme aufgrund von Nichtverfügbarkeit von TIA Openness

Die CAX-Implementierung basiert auf TIA Openness Public APIs. Openness Public APIs sind nur verfügbar, wenn der Anwender das Openness-Optionspaket während der Installation des TIA Portals installiert hat. Deshalb muss vor der Ausführung von CAX-bezogenen Funktionen geprüft werden, ob Openness verfügbar ist. (Siehe TIA Portal Openness installieren (Seite 40))

Immer wenn der Anwender eine CAX-Exportaktion oder eine CAX-Importaktion in der Benutzeroberfläche des TIA Portals auslöst, wird eine Prüfung durchgeführt, um zu ermitteln, ob TIA Openness im System vorhanden ist. Wenn keine Installation von TIA Openness gefunden wird, wird dem Anwender ein TIA Portal-Dialog mit der folgenden Fehlermeldung angezeigt.



6.5.21 Round-Trip-Geräte und -Module

Voraussetzung

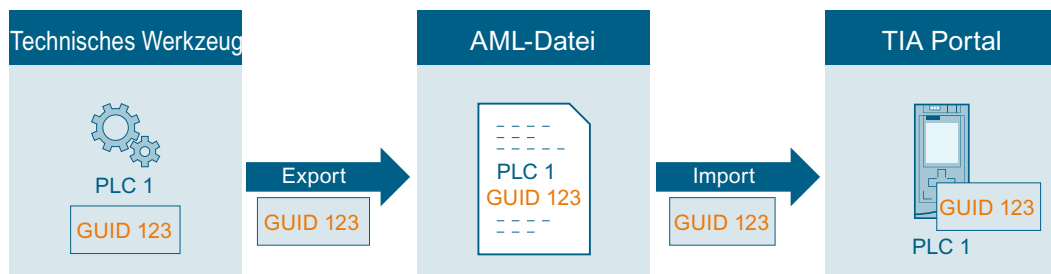
- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist offline.

Verwendung

Sie können zwischen dem TIA Portal und anderen Engineering Tools, beispielsweise einem Elektroplanungs-Tool wie EPLAN oder dem TIA Selection Tool, Konfigurationsdaten austauschen. Zur Identifikation der importierten und exportierten Geräte wird eine eindeutige globale Kennung, die AML GUID, verwendet.

Während der Round-Trips wird die AML GUID für physische Objekte wie Geräte und Geräteelemente, die nicht integriert sind, z. B. CPUs oder Module, stabil gehalten, jedoch nicht für virtuelle Objekte wie Variablen, Kanäle usw.

Während des ersten Exports aus dem TIA Portal wird die AML GUID für ein Gerät oder ein nicht integriertes Geräteelement zufällig generiert, danach jedoch stabil gehalten.

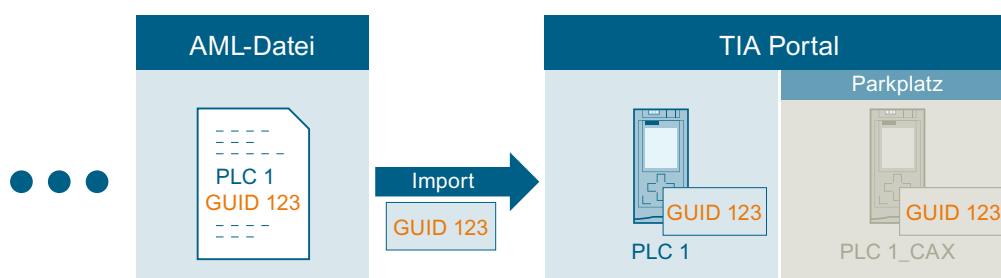


Wenn Sie ein Gerät aus einem technischen Werkzeug in ein leeres TIA Portal-Projekt exportieren, wird der AML GUID in den Kommentar des Hardware-Objekts hinzugefügt. Wenn im TIA Portal unter "Werkzeuge > Einstellungen > CAx > Importeinstellungen" die entsprechende Einstellung aktiviert ist, wird der AML-GUID in der aktuellen Bearbeitungssprache hinzugefügt. Der Round-Trip-Prozess unterstützt nur eine Editorsprache für die AML GUIDs. Wenn Sie Daten importieren oder exportieren, verwenden Sie stets die Bearbeitungssprache, in der Sie den Round-Trip begonnen haben.

Für alle folgenden Importe und Exporte bleibt der AML GUID für dieses Hardware-Objekt gleich. Änderungen am Hardware-Objekt werden übernommen.

Innerhalb eines TIA Portal-Projekts müssen die Objektnamen eindeutig sein. Der Import eines Geräts oder eines Geräteelements in ein TIA Portal-Projekt, in dem ein Objekt mit dem gleichen Namen bereits vorhanden ist, würde zu einem Namenskonflikt führen. Während des Imports haben Sie die Möglichkeit, die Objekte mit den Namenskonflikten in den benutzerdefinierten Parkplatz zu verschieben. Der Name des importierten Objekts wird um "_CAX" erweitert.

Beim CAx-Export der GUI



Um den AML-Round-Trip für Vorgänger-Projektversionen zu unterstützen, werden AML-GUIDs beim Projekt-Upgrade jetzt unter "CustomIdentity" (App-ID) gespeichert und nicht mehr als "Comment" des Geräts/Geräteelements. Die AML-GUIDs des Geräts/Geräteelements müssen eindeutig sein und können in jeder beliebigen Bearbeitungssprache gespeichert werden. Es wird somit davon ausgegangen, dass alle Kommentarbearbeitungssprachen beim Projekt-Upgrade den eindeutigen AML-GUID erhalten. Der AML-GUID des Geräts/Geräteelements ist eindeutig, doch wenn ein neuer GUID mit passendem [AR_APC:ID:*]-Regex in einer beliebigen Bearbeitungssprache dem Kommentar hinzugefügt wird, gilt der erste GUID, der aus den Bearbeitungssprachen ausgewählt wird, als der AML-GUID für das Gerät/Geräteelement.

Sobald die AML-GUIDs im Kommentar in CustomIdentity verschoben werden, wird der Kommentar zum Gerät/Geräteelement durch Entfernen von [AR_APC:ID:*] aktualisiert. Das wird im folgenden Bild gezeigt.

Wenn der Kommentar mehrere Abschnitte [AR_APC:ID:*] enthält, wird der erste GUID, der dem Muster entspricht, auf das CustomIdentity-Repository gesetzt und aus dem Kommentar entfernt. Der restliche Text wird als Kommentar betrachtet.

Name:	PLC_2
Author:	IC660850
Comment:	TestCPU

Hinweis

Kopieren eines importierten Geräts

Wenn Sie ein Gerät oder ein Geräteelement mit einem AML-GUID kopieren, müssen Sie den AML-GUID im Kommentar des kopierten Objekts löschen. Andernfalls existieren Geräte oder Geräteelemente mit identischen AML GUIDs in Ihrem Projekt und führen zu einer ungültigen AML-Datei.

Importeinstellungen

1. Legen Sie den Namen des Parkplatzordners unter "Optionen > Einstellungen > CAX > Einstellungen für Konfliktlösung" fest.
Der Parkplatzordner dient zum Speichern von Objekten mit Namenskonflikten.
2. Aktivieren Sie "Optionen > Einstellungen > CAX > Importeinstellungen > GUIDs beim Import speichern".

Beim Import:

- Die GUIDs eines physischen Asset werden als Bestandteil des CustomIdentity-Abschnitts gespeichert.
- Beim Importieren einer AML-Datei werden die GUIDs als Bestandteil der CustomIdentity gespeichert.

Hinweis

Gültige AML-GUIDs

Wenn Sie einen AML-GUID vor dem Import bearbeiten, wird der AML-GUID ungültig und der CAX-Import wird abgebrochen und entsprechende Informationen werden protokolliert.

Hinweis

Kommentar zu lang

Sollte der Anhang des AML GUID Kommentars die maximale Länge von 500 Zeichen überschreiten, wird der Benutzerkommentar auf 500 Zeichen begrenzt. Eine entsprechende Information wird protokolliert.

Exportieren

Beim Export:

- Der zu exportierende GUID wird für den Schlüssel AR_APC:ID aus CustomIdentity exportiert.
- Wenn der GUID nicht als Bestandteil von CustomIdentity verfügbar ist, wird der Mehrbenutzer-GUID abgerufen und als AML GUID exportiert.
- Wenn es durch keinen der beiden obigen Schritte gelingt, den GUID eines physischen Asset bereitzustellen, wird vom Exportvorgang ein neuer zufälliger GUID erzeugt, der als AML-GUID für physische Assets behandelt wird.

AML-Struktur

Die erstellte Kennung wird wie in dem folgenden Code-Ausschnitt beschrieben in eine AML Datei exportiert:

```
<InternalElement ID="23aeefd0-ce05-4116-a644-e33d43901eaf"  
Name="PLC_1"
```

6.5.22 Export/Import-Topologie

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist offline.

Verwendung

Im TIA Portal können Sie die Geräte mit ihren Topologieinformationen in eine AML-Datei exportieren. Beim Importieren in ein leeres TIA Portal-Projekt behalten die importierten Geräteelemente die Topologieinformationen bei.

Das Element <InternalLink> liefert Verknüpfungsdetails von Port-zu-Port-Verschaltungen zwischen den Geräteelementen. Es erscheint unter dem gemeinsamen übergeordneten Objekt der verschalteten Geräte und enthält eindeutige Variablennamen.

Attribute eines Elements "InternalLink"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	Die Variablennamen sind als "Link to Port_n" formatiert (dabei steht n für eine Zahl von 1 bis zu der Anzahl von Port-zu-Port-Verschaltungen).
RefPartnerSideA	Obligatorisch	Bezeichnet den verknüpften Port. Formatiert als UniqueIDOfPort:CommunicationPortInterface
RefPartnerSideB	Obligatorisch	Bezeichnet den verknüpften Port. Formatiert als UniqueIDOfPort:CommunicationPortInterface

Beispiel: Topologiesicht



AML-Struktur

Die folgende Abbildung zeigt eine partielle Elementstruktur der exportierten AML-Datei. Sie enthält zwei eindeutige IDs für die Ports in PLCs.

```
...
<InternalElement ID="e1966b52-b8b3-47b4-8866-a754ebb77648" Name="Port_1">
  <Attribute Name="Label" AttributeDataType="xs:string">
    ...
  ...
</InternalElement>
...
<InternalElement ID="75f31daf-575f-48a2-ab35-8f07a376eb1b" Name="Port_1">
  <Attribute Name="Label" AttributeDataType="xs:string">
    ...
  ...
</InternalElement>
```

Das Element <Interallink> enthält drei obligatorische Attribute.

```
<Internallink Name="Link to Port_1"
  RefPartnerSideA="e1966b52-b8b3-47b4-8866-a754ebb77648:CommunicationPortInterface"
  RefPartnerSideB="75f31daf-575f-48a2-ab35-8f07a376eb1b:CommunicationPortInterface" />
```

6.5.23 Import eines Geräts mit Bibliotheksreferenzen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Sie können Geräte (Stationen)/Geräteelemente (Module, Submodule) in einer AML-Datei mit Hilfe von Bibliotheksreferenzen in TIA Portal importieren.

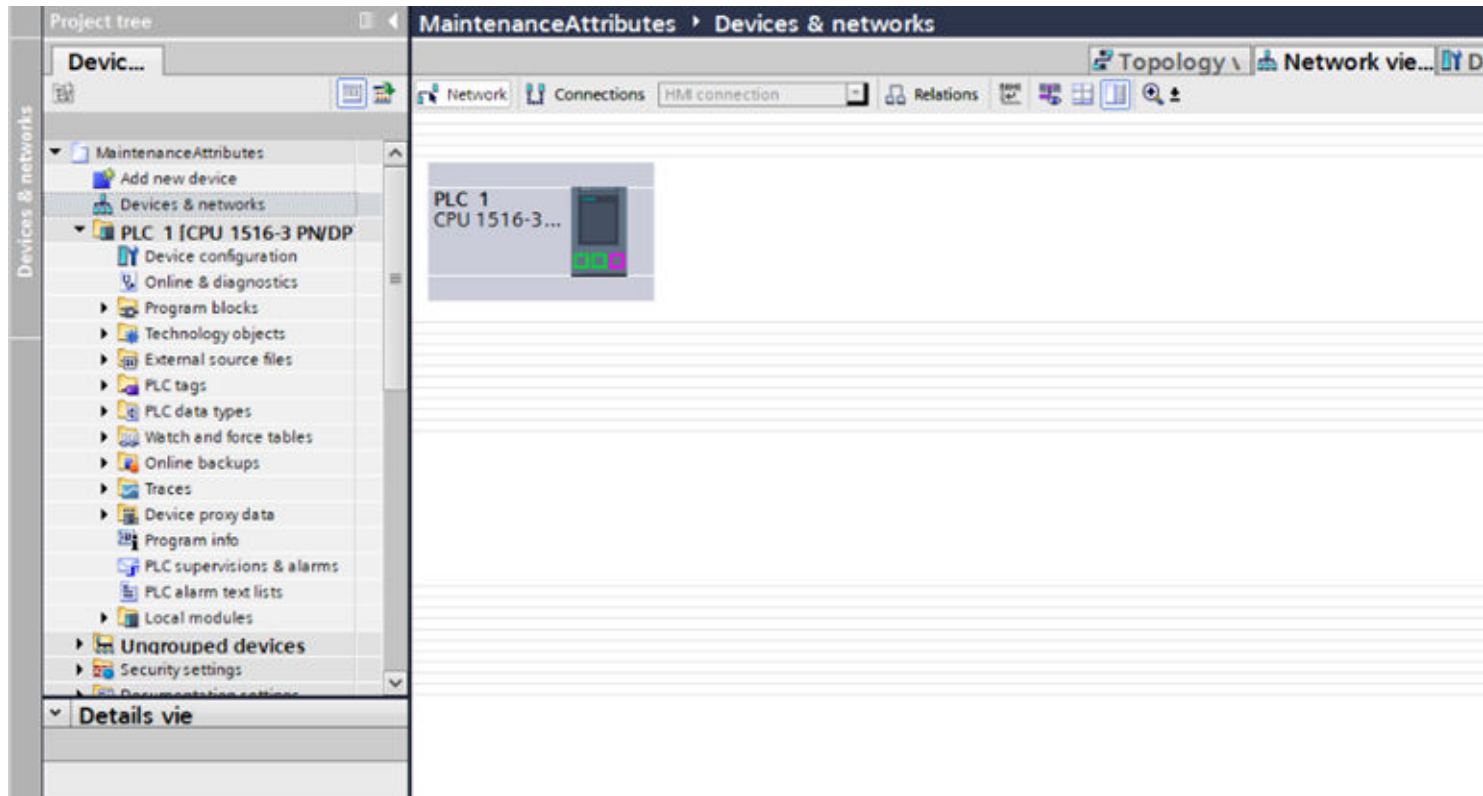
AML-Struktur

Die folgende AML-Struktur beschreibt die XML-Datei, die während des Imports in ein TIA Portal-Projekt zu verwenden ist.

```
...
<InternalElement ID="bed34f88-7a3f-4e37-a32f-dfla6dcb954a" Name="S71500/ET200MP station_1">
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>System:Device.S71500</Value>
<Attribute Name="TemplateIdentifier" AttributeDataType="xs:string">
<Value>GlobalLib://StationPlcLibrary/Master copies/DeviceFolder/S71500ET200MP station_1</
Value>
</Attribute>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
Device" />
...
<InternalElement ID="5082f437-4198-4dcb-b794-35b6b9fcd104" Name="PLC_1">
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 214-1BE30-0XB0</Value>
<Attribute Name="TemplateIdentifier" AttributeDataType="xs:string">
<Value>GlobalLib://StationPlcLibrary/Master copies/PLC_1</Value>
</Attribute>
</Attribute>
</InternalElement>
...
</InternalElement>
...
```

Beispiel: Importierte Konfiguration

Obiger Abschnitt der AML-Datei entspricht der Station und dem PLC im TIA Portal wie unten dargestellt:



Bei dieser Funktion sind folgende Punkte zu beachten:

- Mit dieser Funktion wird die Verwendung von Bibliotheken auf globale Bibliotheken beschränkt.
- Besitzt ein Gerät/Geräteelement ein Attribut TemplatelIdentifier, hat dieses Priorität vor Type Identifier.
- Es wird vorausgesetzt, dass die beim Import verwendeten globalen Bibliotheken vor dem Start des Imports geladen wurden, andernfalls wird eine entsprechende Fehlermeldung ausgegeben.
- Für den Import ist aus allen Systemordnern in einer Bibliothek nur der Ordner 'Master Copies' zu berücksichtigen.
- Zur Unterstützung der Mehrsprachigkeit des TIA Portals kann der TemplatelIdentifier im Pfad Master Copy in allen von TIA unterstützten Sprachen abgelegt sein, jedoch sollte "GlobalLib:/" vorangestellt werden. Der Benutzer von TIA Portal muss ferner sicherstellen, dass die TIA-Oberflächensprache und die Sprache der AML-Datei übereinstimmen.
- Für einen fehlerfreien Import dürfen die Masterkopien des Geräts/Geräteelements in der Bibliothek und das Gerät/Geräteelement in der AML-Datei keine Elemente enthalten, bei denen Konflikte auftreten. Konflikte können hier zum Datenverlust führen.

- Wenn das Gerät/Geräteelement in der AML-Datei eine allgemeine Typkennung oder überhaupt keine Typkennung hat, wird der normale Ablauf für den Ersatz der Typkennung eingehalten. Das Gerät/Geräteelement muss jedoch aus dem Templatelidentifizier erstellt werden, und nach der Erstellung darf die Typkennung nicht mit der ersetzten Typkennung abgeglichen werden.
- In der AML-Datei kann der Templatelidentifizier unter dem Attribut Typelidentifizier auf jedem steckbaren Element und auch nur auf steckbaren Elementen erscheinen. Jede andere Einheit in der AML-Datei (z.B. Kanal/Variable) muss, wenn sie eine Template Reference aufweist, als ungültig behandelt werden und es ist eine entsprechende Meldung für den Benutzer zu erzeugen.
- Das Zeichen '/' im Pfad Templatelidentifizier ist ein Trennzeichen für die Trennung von Ordnern. Wenn ein Masterkopieobjekt einen Namen hat, der das Zeichen '/' enthält, kann dies dazu führen, dass CAX das Masterkopieobjekt nicht findet.
- Zusammenführungen sollten beim Import hinsichtlich der Gerätekonfiguration mit Templatelidentifizier genauso wie bei einer Gerätekonfiguration mit Typelidentifizier funktionieren.
- Bestehende Zusammenführungen mit einer AML-Datei nur mit Typelidentifizier sollten sich ebenfalls normal verhalten.
- Bei einem Konflikt, bei dem ein Gerät sowohl in TIA als auch in der AML-Datei vorhanden ist (mit einem Bibliotheksverweis darauf) und im Gerät in der Bibliothek die PLC integriert ist, kann beim Zusammenführen nur die Station, aber nicht die PLC umbenannt werden.
- Die Einstellung der Anfangsadresse für Ein-/Ausgabebaugruppen, wenn diese aus der Masterkopie importiert werden, unterliegt einer Einschränkung. Sobald eine Anfangsadresse einer Baugruppe bereits zugewiesen ist, kann diese Anfangsadresse nicht mehr eingestellt werden, wenn sie während des Imports aus Templatelidentifizier erstellt wird. Zu beachten ist hier, dass der Benutzer die Baugruppe in der Masterkopie hinzufügen muss (bei der Einrichtung globaler Bibliotheken), bevor ihr eine Adresse zugewiesen wird; die Baugruppe sollte also nicht mit ihrem Master-Controller verbunden sein, wenn sie in die Masterkopie eingefügt wird.

Hinweis

- Eine AML-Datei für V16/V17 mit Gerät/Geräteelement und 'Templatelidentifizier' sollte erfolgreich in TIA Portal V17 importiert werden
 - Eine AML-Datei für V16/V17 mit Gerät/Geräteelement und 'TemplateReference' sollte nicht erfolgreich in TIA Portal V17 importiert werden
 - Eine AML-Datei für V16 mit Gerät/Geräteelement und 'TemplateReference' sollte erfolgreich in V16 importiert werden. Dieses Verhalten für V16 TIAP darf nicht geändert werden.
-
- Vor TIA Portal V17 konnte eine Gerätekonfiguration, bei der im TIA-Portal kein Drag & Drop in Bibliotheken auf Modulebene (vor allem von Hauptmodulen wie Kopfmodul/CPU) möglich war, in CAX-Bibliotheksworkflows nicht unterstützt werden. Zu den jetzigen Konfigurationsmöglichkeiten gehört jedoch uneingeschränktes Drag & Drop in Bibliotheken auf Geräteebene, bei dem der Benutzer einen vorkonfigurierten Baugruppenträger und ein vorkonfiguriertes Hauptmodul als Teil der Masterkopie eines Geräts auswählt und diese in die Bibliothek einstellt.

Aufgrund dieser Unterstützung durch TIA Portal V17 umfasst der CAx-Workflow jetzt auch den Import der gesamten Geräte-Masterkopie aus der Bibliothek, in der sich bereits der vorkonfigurierte Baugruppenträger und das vorkonfigurierte Hauptmodul befinden

Unten finden Sie einige Beispiele für das erwartete Verhalten beim Importieren einer vorab definierten Geräte-Masterkopie mit Hilfe einer AML-Datei.

- In TIA Portal V17 müsste jede AML-Datei mit einer Gerätekonfiguration, die Bibliotheksverweise auf Geräteebene (einschließlich Baugruppenträger und Hauptmodul) beinhaltet, einen erfolgreichen Import dadurch ermöglichen, dass ein Gerät anhand der Bibliotheksdaten erstellt wird, wobei der Baugruppenträger und das Hauptmodul ebenfalls aus der Geräte-Masterkopie übernommen werden.
 - Voraussetzung hierfür: Der Bibliotheksverweis ist gültig
 - Weitere Voraussetzung: Typenidentifizierer von Baugruppenträger und Hauptmodul in der Bibliothek stimmt mit dem entsprechenden AML-Typenidentifizierer überein
- In TIA Portal V17 müsste jede AML-Datei mit einer Gerätekonfiguration, die Bibliotheksverweise auf Geräteebene (einschließlich Baugruppenträger und Hauptmodul) beinhaltet, zu einem Import mit Fehler dadurch führen, dass ein Gerät trotzdem anhand der Bibliotheksdaten erstellt wird, wobei der Baugruppenträger und das Hauptmodul ebenfalls aus der Geräte-Masterkopie übernommen werden
 - Voraussetzung: Der Bibliotheksverweis ist gültig
 - Weitere Voraussetzung: Typenidentifizierer von Baugruppenträger und Hauptmodul in der Bibliothek stimmt nicht mit dem entsprechenden AML-Typenidentifizierer überein
- In TIA Portal V17 müsste jede AML-Datei mit einer Gerätekonfiguration, die Bibliotheksverweise auf Geräteebene (einschließlich Baugruppenträger und Hauptmodul) beinhaltet, zu einem Import ohne Warnung dadurch führen, dass ein Gerät trotzdem anhand der Bibliotheksdaten erstellt wird, wobei der Baugruppenträger und das Hauptmodul ebenfalls aus der Geräte-Masterkopie übernommen werden.
 - Voraussetzung: Der Bibliotheksverweis ist gültig
 - Weitere Voraussetzung: FW von Baugruppenträger oder Hauptmodul in der Geräte-Masterkopie stimmt nicht mit der entsprechenden AML-FW überein
- In TIA Portal V17 müsste jede AML-Datei mit einer Gerätekonfiguration, die Bibliotheksverweise auf Geräteebene (einschließlich Baugruppenträger und Hauptmodul) beinhaltet, zu einem Import ohne Warnung dadurch führen, dass ein Gerät trotzdem anhand der Bibliotheksdaten erstellt wird, wobei der Baugruppenträger und das Hauptmodul ebenfalls aus der Geräte-Masterkopie übernommen werden.
 - Voraussetzung: Der Bibliotheksverweis ist gültig
 - Weitere Voraussetzung: FW von Baugruppenträger oder Hauptmodul fehlt in AML

Empfehlung für den Einsatz von Bibliotheksverweisen

Im Allgemeinen unterstützt CAx mehrere Arten der Behandlung von Bibliotheksverweisen beim CAx-Import. Der Benutzer kann mit jedem der nachstehend beschriebenen Verfahren erreichen, dass eine AML-Datei erfolgreich importiert wird. Beispiel:

1. Wenn das TIA Portal einen Bibliotheksworkflow (d. h. Verwendung von Drag & Drop) in Bezug auf das Hauptmodul (z. B. Kopfmodul/CPU) unterstützt, bei dem dieses in der Gerätekonfiguration als "eigenständige" Masterkopie vorliegt, muss der CAx-Benutzer die AML-Datei so konfigurieren, dass die Geräte/Geräteelemente über "eigenständige" Bibliotheksverweise verfügen. Hierbei wird davon ausgegangen, dass das Hauptmodul in der Bibliothek als eigenständige Masterkopie vorliegt.
2. Wenn das TIA Portal keinen Bibliotheksworkflow (d. h. Verwendung von Drag & Drop) in Bezug auf das Hauptmodul (z. B. Kopfmodul/CPU) unterstützt, bei dem dieses in der Gerätekonfiguration als "eigenständige" Masterkopie vorliegt, muss der CAx-Benutzer die AML-Datei so konfigurieren, dass das Geräteelement über Vorlagen-IDs verfügt. Hierbei wird davon ausgegangen, dass das Hauptmodul als Teil einer vollständigen Geräte-Masterkopie in der Bibliothek vorliegt.

Hinweis

Jede Gerätekonfiguration weist in Bezug auf Bibliotheksworkflows (Drag & Drop) im TIA Portal ein anderes Verhalten auf. Manche Gerätefamilien unterstützen Bibliotheksworkflows auf Detailebene (Geräteelement-Ebene), manche nur auf der Ebene vollständiger Geräte und manche auf beiden Ebenen. Ein CAx-Benutzer muss daher die Bibliotheksverweise in AML-Dateien so konfigurieren, dass diese Aspekte des TIA-Portals berücksichtigt werden.

Whitelist und Blacklist

- Wenn ein Gerät/Geräteelement in der Whitelist bzw. Blacklist aufgeführt ist, bedeutet dies Folgendes:
 - Entweder wurde das Gerät/Geräteelement als "eigenständige" Masterkopie ohne untergeordnete Geräteelemente bestätigt;
 - Oder die Gerätekonfiguration wurde als "vollständige" Geräte-Masterkopie mit darin enthaltenem Baugruppenträger und Hauptmodul bestätigt.
- Ein Geräteelement kann in mehreren Gerätekonfigurationen zugleich verwendet werden. Dass ein Geräteelement in der Whitelist aufgeführt ist, bedeutet nicht zwangsläufig, dass der Bibliotheksimport des Geräteelements in allen mit dem Element möglichen Konfigurationen funktioniert.
 - Beispiel: Wenn ein Geräteelement - x mit der Firmwareversion - y aufgeführt ist, kann trotzdem Folgendes eintreten:
 - Es funktioniert unter Umständen nicht mit anderen Firmwareversionen.
 - Es funktioniert unter Umständen nicht, wenn es in ein anderes Modul bzw. eine andere Station gesteckt wird.

Die nachstehende Tabelle zeigt die Liste der Geräte/Geräteelemente, die den Import mit Hilfe von Bibliotheksverweisen unterstützen.

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 212-1HD30-0XB0	
OrderNumber:6ES7 510-1DJ00-0AB0	
OrderNumber:6ES7 221-3BD30-0XB0	
OrderNumber:6ES7 511-1CK00-0AB0	V2.2
OrderNumber:6ES7 516-3AN01-0AB0	V1.8
OrderNumber:6ES7 518-4AP00-0AB0	V1.0
OrderNumber:6GK7 542-5DX00-0XE0	V1.8
OrderNumber:6ES7 611-4SB00-0YB7	V1.8
OrderNumber:6ES7 315-2FJ14-0AB0	V1.8
OrderNumber:6ES7 154-8AB01-0AB0	V1.0
OrderNumber:6ES7 141-4BF00-0AA0	V4.6
OrderNumber:6ES7 137-6BD00-0BA0	V3.1
OrderNumber:6ES7 147-4JD00-0AB0	V3.2
OrderNumber:6ES7 154-4AB10-0AB0	
OrderNumber:3RK7 137-6SA00-0BC1	V2.2
OrderNumber:6ES7 518-4FP00-0AB0	V1.0
OrderNumber:6GK7 543-1AX00-0XE0	V7.0/Cu
OrderNumber:6ES7 516-2GN00-0AB0	V1.1
OrderNumber:6ES7 148-4CA00-0AA0	V2.5
OrderNumber:6ES7 148-4CA60-0AA0	V2.1
OrderNumber:6ES7 154-8FX00-0AB0	V2.1
OrderNumber:6ES7 148-4EA00-0AA0	
OrderNumber:6ES7 151-7AA20-0AB0	
OrderNumber:6ES7 138-4CA50-0AB0	V3.2
OrderNumber:6ES7 151-8AB00-0AB0	
OrderNumber:6ES7 138-4CB11-0AB0	V2.6
OrderNumber:6ES7 151-7AA21-0AB0	
OrderNumber:6ES7 138-4CA60-0AB0	V2.7
OrderNumber:6ES7 151-7FA21-0AB0	
OrderNumber:6ES7 138-4CA01-0AA0	V3.3
OrderNumber:6GT2002-0HD00	
OrderNumber:6ES7 142-4BD00-0AA0	V3.3
OrderNumber:6SL3 235-0TE21-1RB0	
OrderNumber:6SL3 235-0TE21-1SB0	V1.0
OrderNumber:6SL3 514-1KE13-5AE0	
OrderNumber:6ES7 148-4EB00-0AA0	V3.1
OrderNumber:6ES7 143-4BF00-0AA0	V3.1
OrderNumber:6ES7 134-6JD00-0CA1	V4.7.6
OrderNumber:6ES7 193-6PA00-0AA0	
OrderNumber:6ES7 512-1SK00-0AB0	
OrderNumber:3RK1 308-0AB00-0CP0	V2.0
OrderNumber:6ES7 512-1DK00-0AB0	V1.1

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 137-6BD00-0BA0	V1.8
OrderNumber:6ES7 135-6HD00-0BA1	V1.1
OrderNumber:6ES7 137-6AA00-0BA0	V1.8
OrderNumber:6ES7 510-1SJ01-0AB0	V2.1
OrderNumber:6ES7 138-6AA00-0BA0	V1.1
OrderNumber:3RK1 308-0BE00-0CP0	V1.0
OrderNumber:6ES7 135-6HB00-0DA1	V2.1
OrderNumber:6ES7 134-6FF00-0AA1	V1.2
OrderNumber:3RK1 308-0AE00-0CP0	V1.1
OrderNumber:6ES7 390-1***0-0AA0	V2.0
OrderNumber:6ES7 307-1BA00-0AA0	V1.0
OrderNumber:6ES7 318-3FL01-0AB0	V1.1
OrderNumber:6ES7 338-7XF00-0AB0	
OrderNumber:6ES7 307-1KA01-0AA0	
OrderNumber:6ES7 317-2FK13-0AB0	V3.2
OrderNumber:6GT2002-0GA10	
OrderNumber:6ES7 307-1EA00-0AA0	
OrderNumber:6ES7 317-6FF04-0AB0	V2.6
OrderNumber:6ES7 323-1BH01-0AA0	
OrderNumber:6ES7 307-1EA80-0AA0	
OrderNumber:6ES7 315-2FJ14-0AB0	V3.3
OrderNumber:6ES7 350-1AH03-0AE0	
OrderNumber:6ES7 307-1EA01-0AA0	
OrderNumber:6ES7 318-3EL00-0AB0	V3.2
OrderNumber:6ES7 321-7EH00-0AB0	
OrderNumber:6ES7 307-1KA02-0AA0	
OrderNumber:6ES7 317-2EK13-0AB0	V2.8
OrderNumber:6ES7 323-1BL00-0AA0	
OrderNumber:6ES7 314-6BG03-0AB0	
OrderNumber:6ES7 340-1AH02-0AE0	V2.6
OrderNumber:6ES7 313-6BG04-0AB0	

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 360-3AA01-0AA0	V2.6
OrderNumber:6ES7 314-1AG13-0AB0	V1.0
OrderNumber:6ES7 321-7TH00-0AB0	V3.3
OrderNumber:3RK1 335-0AS01-0AA0	
OrderNumber:3RK1 305-0AS01-0AA0	V2.6
OrderNumber:3RW4900-0NCO	
OrderNumber:6ES7 518-4AP00-0AB0	V41.0
OrderNumber:6ES7 155-5AA00-0AA0	
OrderNumber:6ES7 523-1BL00-0AA0	V2.0
OrderNumber:6ES7 534-7QE00-0AB0	V2.1
OrderNumber:6ES7 157-1AA00-0AB0	V4.0
OrderNumber:6ES7 144-5KD00-0BA0	V1.0
OrderNumber:6ES7 145-5ND00-0BA0	V1.0
OrderNumber:6ES7 143-5AF00-0BA0	V1.0
OrderNumber:6ES7 147-5JD00-0BA0	V1.0
OrderNumber:6ES7 314-6EH04-0AB0	V1.0
OrderNumber:6ES7 153-4BA00-0XB0	V1.0
OrderNumber:6GK7 343-2AH01-0XA0	V1.0
OrderNumber:6ES7 155-6BU00-0CN0	V3.3
OrderNumber:6ES7 131-6BF00-0BA0	V4.0
OrderNumber:6ES7 516-3AN01-0AB0	V3.1
OrderNumber:6EP1332-4BA00	V3.1
OrderNumber:6EP3436-8MB00-2CY0	V1.1
OrderNumber:6EP4293-8HB00-0XY0	V2.1
OrderNumber:6EP4297-8HB00-0XY0	
OrderNumber:6ES7 212-1BD30-0XB0	V1.1
OrderNumber:6EP4137-3AB00-2AY0	V1.1
OrderNumber:6EP4134-0GB00-0AY0	V1.1
OrderNumber:6ES7 516-3AN00-0AB0	V2.2
OrderNumber:6ES7 400-1TA01-0AA0	V2.2
OrderNumber:6ES7 405-0DA02-0AA0	V1.0
OrderNumber:6ES7 414-3EM05-0AB0	V1.8
OrderNumber:6ES7 412-1XJ05-0AB0	
OrderNumber:6GK7 443-1EX11-0XE0	
OrderNumber:6ES7 312-1AE13-0AB0	V5.3
OrderNumber:6GK7 343-1CX10-0XE0	V5.3
OrderNumber:6GK7 443-5DX04-0XE0	V2.7
OrderNumber:6GK7 342-5DA02-0XE0	V2.6
OrderNumber:6ES7 511-1AK00-0AB0	V2.4
OrderNumber:6GK7 542-1AX00-0XE0	V6.6
OrderNumber:6ES7 211-1BD30-0XB0	V5.0
OrderNumber:6GK7 242-7KX30-0XE0	V1.8

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 414-3EM05-0AB0	V1.0
OrderNumber:6ES7 151-1AA05-0AB0	V2.2
OrderNumber:6ES7 155-5AA00-0AC0	V1.0
OrderNumber:6ES7 155-6BU00-0CN0	V5.2
OrderNumber:6ES7 193-6PA00-0AA0	
OrderNumber:6ES7 505-0KA00-0AB0	V3.0
OrderNumber:6ES7 521-1BH50-0AA0	V3.0
OrderNumber:6ES7 511-1TK01-0AB0	V1.0
OrderNumber:6ES7 515-2TM01-0AB0	V1.0
OrderNumber:6ES7 521-1BH00-0AB0	V2.0
OrderNumber:6ES7 521-1BL10-0AA0	V2.1
OrderNumber:6ES7 521-7EH00-0AB0	V2.1
OrderNumber:6ES7 521-1FH00-0AA0	V2.1
OrderNumber:6ES7 522-5HH00-0AB0	V1.0
OrderNumber:6AG1 511-1AK00-7AB0	V1.0
OrderNumber:6AG1 505-0KA00-7AB0	V2.0
OrderNumber:6AG1 531-7KF00-7AB0	V1.0
OrderNumber:6AG1 531-7NF10-7AB0	V1.8
OrderNumber:6AG1 532-5HD00-7AB0	V1.0
OrderNumber:6AG1 532-5HF00-7AB0	V2.0
OrderNumber:6AG1 521-1BH00-7AB0	V2.0
OrderNumber:6AG1 521-1FH00-7AA0	V2.0
OrderNumber:6AG1 522-1BF00-7AB0	V2.0
OrderNumber:6AG1 522-5FF00-7AB0	V2.0
OrderNumber:6AG1 550-1AA00-7AB0	V2.0
OrderNumber:6ES7 522-1BH01-0AB0	V2.0
OrderNumber:6AG1 513-1AL00-2AB0	V2.0
OrderNumber:6AG1 541-1AB00-7AB0	V1.1
OrderNumber:6AG1 542-5DX00-7XE0	V1.0
OrderNumber:6ES7 505-0RA00-0AB0	V1.8
OrderNumber:6ES7 518-4AP00-3AB0	V1.0
OrderNumber:6GK7 542-1AX00-0XE0	V1.0
OrderNumber:6ES7 551-1AB00-0AB0	V1.0
OrderNumber:6ES7 531-7PF00-0AB0	V2.1
OrderNumber:6AG1 516-3AN00-7AB0	V2.0
OrderNumber:6ES7 517-3TP00-0AB0	V1.1
OrderNumber:6ES7 540-1AD00-0AA0	V1.1
OrderNumber:6ES7 532-5HD00-0AB0	V1.8
OrderNumber:6ES7 507-0RA00-0AB0	V2.1
OrderNumber:6ES7 515-2AM00-0AB0	V1.0
OrderNumber:7MH4 980-1AA01	V2.1
OrderNumber:7MH4 980-2AA01	V1.0
OrderNumber:6ES7 553-1AA00-0AB0	V1.8

Whitelist	
TypIdentifier	FirmwareVersion
OrderNumber:6ES7 552-1AA00-0AB0	
OrderNumber:6ES7 550-1AA00-0AB0	
OrderNumber:6AG1 516-3AN00-2AB0	V1.0
OrderNumber:6AG1 540-1AD00-7AA0	V1.0
OrderNumber:6AG1 540-1AB00-7AA0	V1.1
OrderNumber:6AG1 541-1AD00-7AB0	V1.8
OrderNumber:6ES7 511-1CK00-0AB0	V1.0
OrderNumber:6ES7 531-7NF10-0AB0	V1.0
OrderNumber:6ES7 532-5HF00-0AB0	V1.0
OrderNumber:6ES7 512-1CK00-0AB0	V2.1
OrderNumber:6ES7 521-1BH10-0AA0	V2.1
OrderNumber:6ES7 522-1BF00-0AB0	V2.1
OrderNumber:6ES7 212-1BE40-0XB0	V2.1
OrderNumber:6ES7 234-4HE32-0XB0	V1.0
OrderNumber:6ES7 223-1BH30-0XB0	V2.1
OrderNumber:6GK7 243-1BX30-0XE0	V4.2
OrderNumber:3RK7243-2AA30-0XB0	V2.0
OrderNumber:6ES7 217-1AG40-0XB0	V1.0
OrderNumber:6GT2 002-0LA00	V3.0
OrderNumber:6ES7 518-4FP00-0AB0	V1.1
OrderNumber:6ES7 517-3UP00-0AB0	V4.2
OrderNumber:6ES7 511-1FK00-0AB0	V1.0
OrderNumber:6ES7 531-7QD00-0AB0	V2.1
OrderNumber:6ES7 517-3AP00-0AB0	V2.1
OrderNumber:6ES7 522-5EH00-0AB0	V1.8
OrderNumber:6ES7 522-1BH10-0AA0	V1.0
OrderNumber:3SU1 400-1LK10-*AA1	V2.1
OrderNumber:3SU1 400-1MA10-1BA1	V1.0
OrderNumber:6EP3436-8MB00-2CY0	V1.0
OrderNumber:6EP4436-8XB00-0CY0	V1.0
OrderNumber:6EP4131-0GB00-0AY0	V1.2
GSD:SIEM804C.GSD/M/0	V1.2
OrderNumber:6ES7 143-2BH*0-0AB0	V1.0
OrderNumber:6ES7 143-2BH00-0AB0	
OrderNumber:6ES7 407-0DA02-0AA0	
OrderNumber:6ES7 460-0AA01-0AB0	
OrderNumber:6ES7 412-2XK07-0AB0	
OrderNumber:6ES7 450-1AP00-0AE0	
OrderNumber:6ES7 211-1AE31-0XB0	
OrderNumber:6ES7 241-1CH32-0XB0	V7.0
OrderNumber:6ES7 211-1HD30-0XB0	
OrderNumber:6ES7 241-1AH30-0XB0	V3.0
OrderNumber:6ES7 212-1AE31-0XB0	V2.2

6.5 Hardware-Daten importieren/exportieren

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 221-1BF32-0XB0	V2.2
OrderNumber:6ES7 232-4HD30-0XB0	V1.0
OrderNumber:6ES7 510-1DJ01-0AB0	V3.0
OrderNumber:6ES7 512-1DK01-0AB0	V2.0
OrderNumber:6ES7 157-1AB00-0AB0	V1.0
OrderNumber:6ES7 512-1CK00-0AB0	V1.8
OrderNumber:6AG1 155-6AU00-7BN0	V1.8
OrderNumber:6AG1 193-6PA00-7AA0	V1.0
OrderNumber:6AG1 155-6AU00-7BN0	V1.8
OrderNumber:6ES7 313-6CF03-0AB0	V1.0
OrderNumber:6ES7 318-3EL00-0AB0	V1.0
OrderNumber:6ES7 151-3BA23-0AB0	V3.1
OrderNumber:3RK1 903-0BA00	V2.6
OrderNumber:3RK1 301-0BB13-1AA4	V2.7
OrderNumber:6ES7 518-4AP00-0AB0	V7.0
OrderNumber:6ES7 154-8AB00-0AB0	
OrderNumber:6ES7 144-4PF00-0AB0	
OrderNumber:6ES7 151-1BA02-0AB0	V1.6
OrderNumber:6ES7 131-4BB01-0AB0	V2.5
OrderNumber:6ES7 132-4BB01-0AA0	
OrderNumber:6ES7 131-4BB01-0AA0	
OrderNumber:6ES7 132-4BD00-0AB0	
OrderNumber:6ES7 407-0KA02-0AA0	
OrderNumber:6ES7 401-1DA01-0AA0	
OrderNumber:6ES7 416-3ES06-0AB0	
OrderNumber:6ES7 318-3EL01-0AB0	
OrderNumber:6ES7 138-4DA04-0AB0	
OrderNumber:6ES7 518-4FP00-3AB0	V6.0
OrderNumber:6ES7 317-2AJ10-0AB0	V3.2
OrderNumber:6ES7 132-4HB50-0AB0	V1.0
OrderNumber:6ES7 132-4BB31-0AA0	V2.0
OrderNumber:6ES7 416-3ER05-0AB0	V2.6
OrderNumber:6ES7 151-1AA04-0AB0	
GSD:SIEM818A.GSD/M/70000	
GSD:SIEM818A.GSD/M/13	V5.3
OrderNumber:6ES7 155-6AU00-0CN0	
OrderNumber:6ES7 412-2XJ05-0AB0	
OrderNumber:6ES7 431-7QH00-0AB0	
OrderNumber:6ES7 432-1HF00-0AB0	V3.3
OrderNumber:6ES7 460-1BA01-0AB0	V5.3
OrderNumber:6ES7 416-2XN05-0AB0	
GSD:SIEM818A.GSD/M/18	
OrderNumber:7MH4 138-6AA00-0BA0	

Whitelist	
TypIdentifier	FirmwareVersion
OrderNumber:6ES7 155-6AU00-0BN0	V5.3
OrderNumber:6ES7 513-1AL00-0AB0	
OrderNumber:6ES7 521-1BL00-0AB0	
OrderNumber:6ES7 522-1BL00-0AB0	V3.3
OrderNumber:6ES7 522-1BH00-0AB0	V1.8
OrderNumber:6ES7 540-1AB00-0AA0	V2.1
OrderNumber:6ES7 541-1AB00-0AB0	V2.0
OrderNumber:6GK7 542-5DX00-0XE0	V2.0
OrderNumber:6ES7 317-2EK14-0AB0	V1.0
OrderNumber:6ES7 516-3AN00-0AB0	V1.0
OrderNumber:6GK7 542-5FX00-0XE0	V2.0
OrderNumber:6GK7 543-1AX00-0XE0	V3.2
OrderNumber:6GK7 543-1AX00-0XE0	V1.1
OrderNumber:6ES7 516-3AN00-0AB0	V1.0
OrderNumber:6GK7 242-7KX30-0XE0	V1.0
OrderNumber:6ES7 211-1BE31-0XB0	V1.1
OrderNumber:6GK7 243-1JX30-0XE0	V1.5
OrderNumber:6GK7 243-5DX30-0XE0	V1.3
OrderNumber:6ES7 132-4BB01-0AB0	V3.0
OrderNumber:6ES7 152-1AA00-0AB0	V1.0
OrderNumber:6ES7 132-7GD00-0AB0	V1.0
OrderNumber:6ES7 334-0KE00-0AB0	
OrderNumber:6ES7 151-3BA23-0AB0	V2.0
OrderNumber:OPC Server	
OrderNumber:Application	
OrderNumber:6ES7 214-1BG40-0XB0	V6.0
OrderNumber:6ES7 221-1BH32-0XB0	SW V14 ...
OrderNumber:6ES7 222-1XF30-0XB0	SW V8.1 SP2 ...
OrderNumber:6ES7 223-1QH30-0XB0	V4.2
OrderNumber:6ES7 231-4HF30-0XB0	V2.0
OrderNumber:6ES7 234-4HE30-0XB0	V1.0
OrderNumber:7MH4960-2AA01	V1.0
OrderNumber:6ES7 214-1AG40-0XB0	V1.0
OrderNumber:6ES7 228-1RC52-0AA0	V1.0
OrderNumber:6ES7 278-4BD32-0XB0	
OrderNumber:6ES7 214-1HG31-0XB0	V4.2
OrderNumber:7MH4960-6AA01	V2.2
OrderNumber:7MH4960-4AA01	V2.0
OrderNumber:6ES7 223-1PL30-0XB0	V3.0
OrderNumber:6ES7 215-1BG40-0XB0	
OrderNumber:6ES7 231-4HD30-0XB0	
OrderNumber:6ES7 231-5PD32-0XB0	V1.0
OrderNumber:6ES7 231-5QF30-0XB0	V4.2

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 232-4HB32-0XB0 OrderNumber:3RK1 301-OCB10-1AB4	V1.0
	V2.0
OrderNumber:3RK1 903-0CK00	V1.0
OrderNumber:6ES7 215-1HF40-0XB0	V2.0
OrderNumber:6ES7 212-1AF40-0XB0	V10.0
OrderNumber:6ES7 214-1HF40-0XB0	
OrderNumber:6ES7 214-1AF40-0XB0	V4.2
OrderNumber:6ES7 515-2FM00-0AB0	V4.2
OrderNumber:6ES7 511-1UK01-0AB0	V4.2
OrderNumber:6ES7 215-1AG31-0XB0	V4.2
OrderNumber:6ES7 221-1BF30-0XB0	V1.8
OrderNumber:6ES7 222-1XF32-0XB0	V2.1
OrderNumber:6ES7 222-1HF30-0XB0	V3.0
OrderNumber:6ES7 223-1BL30-0XB0	V1.0
OrderNumber:6ES7 215-1HG40-0XB0	V2.0
OrderNumber:6ES7 241-1CH31-0XB0	V1.0
OrderNumber:6ES7 212-1HF40-0XB0	V1.0
OrderNumber:6AG1 214-1AG40-4XB0	V4.2
OrderNumber:6AG1 221-1BF32-2XB0	V1.0
OrderNumber:6AG1 221-1BF32-4XB0	V4.2
OrderNumber:6AG1 222-1BH32-2XB0	V4.2
OrderNumber:6AG1 222-1BF32-4XB0	V2.0
OrderNumber:6AG1 223-1BL32-2XB0	V2.0
OrderNumber:6AG1 214-1HG40-2XB0	V2.0
OrderNumber:6AG1 223-1PH32-2XB0	V2.0
OrderNumber:6AG1 223-1PL32-4XB0	V2.0
OrderNumber:6AG1 223-1QH32-4XB0	V4.2
OrderNumber:6AG1 222-1XF32-2XB0	V2.0
OrderNumber:6AG1 222-1HF32-4XB0	V2.0
OrderNumber:6AG1 222-1HH32-2XB0	V2.0
OrderNumber:6AG1 215-1BG40-5XB0	V2.0
OrderNumber:6AG1 231-5QF32-4XB0	V2.0
OrderNumber:6AG1 231-4HD32-4XB0	V2.0
OrderNumber:6AG1 222-1BF32-2XB0	V4.2
OrderNumber:6AG1 234-4HE32-2XB0	V2.0
OrderNumber:6AG1 278-4BD32-4XB0	V2.0
OrderNumber:6AG1 215-1AG40-4XB0	V2.0
OrderNumber:6AG1 221-1BH32-2XB0	V2.0
OrderNumber:6AG1 241-1CH32-2XB0	V2.0
OrderNumber:6AG1 215-1HG40-2XB0	V4.2
OrderNumber:6ES7 223-1PL32-0XB0	V2.0
OrderNumber:6ES7 228-1RC51-0AA0	V2.1
OrderNumber:6ES7 238-5XA32-0XB0	V4.2

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 215-1AF40-0XB0	V2.0
OrderNumber:6ES7 223-1BL32-0XB0	V2.2
OrderNumber:6AG1 212-1BE40-2XB0	V2.0
OrderNumber:6AG1 243-5DX30-2XE0	V4.2
OrderNumber:6AG1 212-1HE40-2XB0	V2.0
OrderNumber:6AG1 223-1QH32-2XB0	V4.2
OrderNumber:6EP3436-8SB00-2AY0	V1.3
OrderNumber:6EP3437-8MB00-2CY0	V4.2
OrderNumber:6EP4137-3AB00-2AY0	V2.0
OrderNumber:6EP4136-3AB00-2AY0	V1.1
OrderNumber:6EP4133-0GB00-0AY0	V1.1
OrderNumber:6ES7 138-4DC01-0AB0	V2.1
System:Port.Scalance/Comboport_MAU	V2.1
OrderNumber:6GK1 160-4AT01	V1.0
OrderNumber:6GK1 160-4AA00	
OrderNumber:6GK1 161-6AA02	
OrderNumber:6GK1 551-2AA00	V2.7
OrderNumber:6GK1 571-1AA00	V2.6
OrderNumber:6GK1 562-3AA00	V2.7
OrderNumber:6GK1 561-3AA02	SW V6.1 ...
OrderNumber:6GK7 343-1GX31-0XE0	SW V7.1 SP1 ...
OrderNumber:6GK7 443-5DX05-0XE0	SW V7.1 SP2 ...
OrderNumber:6ES7 214-1BE30-0XB0	SW V12 ...
OrderNumber:6ES7 241-1AH32-0XB0	V3.0
OrderNumber:6ES7 211-1AD30-0XB0	V7.0
OrderNumber:6ES7 231-4HA30-0XB0	V2.2
OrderNumber:6ES7 214-1HG40-0XB0	V2.2
OrderNumber:6GK7 242-7KX30-0XE0	V2.2
OrderNumber:6ES7 212-1AD30-0XB0	V2.0
OrderNumber:6ES7 138-4GA50-0AB0	V4.0
OrderNumber:6ES7 132-4BF00-0AB0	V1.4
OrderNumber:6ES7 154-4AB10-0AB0	V2.0
OrderNumber:6ES7 154-8FB01-0AB0	
OrderNumber:6ES7 155-6AU01-0BN0	
OrderNumber:6ES7 155-6AU00-0DN0	V7.1/Cu
OrderNumber:6ES7 155-5AA00-0AB0	V3.2
OrderNumber:6ES7 155-5AA01-0AB0	V4.1
OrderNumber:6AG1 155-5AA00-2AC0	V4.0
OrderNumber:6ES7 155-5BA00-0AB0	V3.0
OrderNumber:6AG1 155-5BA00-2AB0	V4.1
OrderNumber:6ES7 518-4AP00-0AB0	V3.0
OrderNumber:6ES7 132-6BF00-0AA0	V3.0
OrderNumber:6ES7 132-6HD00-0BB0	V3.0

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 132-6BD20-0CA0	V2.5
OrderNumber:6ES7 134-6PA01-0BD0	V1.0
OrderNumber:6ES7 132-6HD00-0BB1	V1.1
OrderNumber:6ES7 531-7NF00-0AB0	V2.0
OrderNumber:6ES7 223-0BD30-0XB0	V3.0
OrderNumber:6ES7 212-1AE40-0XB0	V1.1
OrderNumber:6ES7 221-3AD30-0XB0	V1.1
OrderNumber:6ES7 214-1AE30-0XB0	V1.0
OrderNumber:6ES7 222-1BD30-0XB0	V4.2
OrderNumber:6ES7 222-1AD30-0XB0	V1.0
OrderNumber:6ES7 232-4HA30-0XB0	V2.2
OrderNumber:6ES7 416-3FR05-0AB0	V1.0
OrderNumber:6ES7 964-2AA04-0AB0	V1.0
OrderNumber:6ES7 414-3FM07-0AB0	V1.0
OrderNumber:6ES7 416-2FP07-0AB0	V5.3
OrderNumber:6ES7 151-8FB01-0AB0	
OrderNumber:6ES7 151-7FA20-0AB0	V7.0
OrderNumber:6ES7 132-7RD22-0AB0	V7.0
OrderNumber:6ES7 132-7GD21-0AB0	V3.2
OrderNumber:6ES7 315-2EH13-0AB0	V2.6
OrderNumber:6ES7 322-8BH10-0AB0	
OrderNumber:7MH4 900-2AA01	
OrderNumber:7MH4 900-3AA01	V2.6
OrderNumber:7MH4 950-1AA01	
OrderNumber:7MH4 950-2AA01	
OrderNumber:7MH4920-0AA01	
OrderNumber:7MH4910-0AA01	
OrderNumber:6AG1 155-5AA00-7AB0	
OrderNumber:6AG1 505-0RA00-7AB0	
OrderNumber:6AG1 522-1BL00-7AB0	
OrderNumber:3SU1 401-1MC*0-1CA1	V3.0
OrderNumber:3SU1 401-1ME*0-1DA1	V1.0
OrderNumber:6BK1900-0AA00-0AA*	V1.0
OrderNumber:6BK1900-0BA00-0AA*	V1.0
OrderNumber:6BK1900-0CA00-0AA*	V1.0
OrderNumber:6BK1942-2AA00-0AA*	/HCS4200
OrderNumber:6ES7 412-2EK06-0AB0	/HCS4200
OrderNumber:6GK7 343-2AH11-0XA0	/HCS4200
OrderNumber:6ES7 313-6CG04-0AB0	
OrderNumber:6GK5 991-2AB00-8AA0	V6.0
OrderNumber:6GK5 992-2VA00-8AA0	V3.1
OrderNumber:6GK5 992-2AS00-8AA0	V3.3
OrderNumber:6GK5 400-8AS00-8AP2	

Whitelist	
TypenIdentifier	FirmwareVersion
OrderNumber:6GK5 905-0PA00	/legacy
OrderNumber:6GK5 991-1AD00-8AA0	
OrderNumber:6GK5 992-1AQ00-8AA0	
OrderNumber:6GK5 907-4PA00	
OrderNumber:6ES7 143-2BH50-0AB0	
OrderNumber:6ES7 405-0KA02-0AA0	
OrderNumber:6ES7 513-1FL01-0AB0	
OrderNumber:6ES7 134-7TD00-0AB0	
OrderNumber:6ES7 516-3FN00-0AB0	
OrderNumber:6ES7 151-8AB01-0AB0	
OrderNumber:3RK1 301-0BB10-1AA4	V2.1
OrderNumber:6ES7 151-1CA00-3BL0	V1.8
OrderNumber:3RK1 301-0CB13-0AA4	V3.2
OrderNumber:6ES7 315-2EH14-0AB0	V3.2
OrderNumber:6ES7 321-1BH02-0AA0	
OrderNumber:6ES7 421-7BH01-0AB0	
OrderNumber:6ES7 138-4HA00-0AB0	
OrderNumber:6SL3 244-0SA00-1AA1	
OrderNumber:6SL3 244-0SA01-1AA1	
OrderNumber:6GT2 002-0ED00	
OrderNumber:6ES7 135-7TD00-0AB0 ES9	
OrderNumber:6ES7 340-1CH02-0AE0	
OrderNumber:6ES7 352-1AH02-0AE0	
OrderNumber:6ES7 400-1TA11-0AA0	V1.0
OrderNumber:6ES7 416-3XR05-0AB0	
OrderNumber:3RK1 301-0AB10-1AA3	
OrderNumber:6ES7 672-7AC01-0YA0	
OrderNumber:6ES7 307-1BA01-0AA0	
OrderNumber:6ES7 351-1AH01-0AE0	
OrderNumber:6ES7 334-0CE01-0AA0	
OrderNumber:6ES7 312-5BF04-0AB0	
OrderNumber:6ES7 313-5BG04-0AB0	
OrderNumber:6ES7 338-7XF00-0AB0 IDENT	
OrderNumber:6ES7 314-6CH04-0AB0	V3.3
OrderNumber:6ES7 315-2AH14-0AB0	
OrderNumber:7ME4 120-2DH21-0EA0	
OrderNumber:6ES7 322-1CF00-0AA0	
OrderNumber:6ES7 335-7HG02-0AB0	
OrderNumber:6ES7 338-4BC01-0AB0	
OrderNumber:6ES7 355-2CH00-0AE0	
OrderNumber:7ME4 120-2DH20-0EA0	
OrderNumber:6ES7 321-1CH20-0AA0	
OrderNumber:6ES7 321-1CH00-0AA0	V3.0

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 322-8BF00-0AB0	V1.0
OrderNumber:6ES7 322-1BL00-0AA0	
OrderNumber:6ES7 322-1FF01-0AA0	
OrderNumber:6ES7 305-1BA80-0AA0	
OrderNumber:6ES7 341-1BH01-0AE0	
OrderNumber:6ES7 331-7KB02-0AB0	
OrderNumber:6ES7 331-7KF02-0AB0	
OrderNumber:6ES7 331-1KF01-0AB0	
OrderNumber:6ES7 332-5HB01-0AB0	
OrderNumber:6ES7 332-5HD01-0AB0	
OrderNumber:6ES7 332-5HF00-0AB0	
OrderNumber:6ES7 321-7RD00-0AB0	
OrderNumber:6ES7 321-1FH00-0AA0	
OrderNumber:6ES7 322-1HH01-0AA0	
OrderNumber:6ES7 322-5HF00-0AB0	
OrderNumber:6ES7 322-1HF10-0AA0	
OrderNumber:6ES7 327-1BH00-0AB0	
OrderNumber:6ES7 351-1AH02-0AE0	
OrderNumber:6ES7 321-7BH01-0AB0	
OrderNumber:6ES7 322-1BH01-0AA0	
OrderNumber:6ES7 322-5GH00-0AB0	V1.0
OrderNumber:6ES7 321-1FF01-0AA0	
OrderNumber:6GK7 443-1EX41-0XE0	V5.3
OrderNumber:6ES7 414-2XK05-0AB0	
OrderNumber:6ES7 431-1KF00-0AB0	V5.3
OrderNumber:6ES7 414-3XM05-0AB0	
OrderNumber:6ES7 421-1EL00-0AA0	V7.0
OrderNumber:6ES7 414-3EM07-0AB0	
OrderNumber:6ES7 416-3XS07-0AB0	V7.0
OrderNumber:6ES7 414-3FM06-0AB0	
OrderNumber:6ES7 421-1FH20-0AA0	V6.0
OrderNumber:6ES7 416-3FS07-0AB0	
OrderNumber:6ES7 431-0HH00-0AB0	V7.0
OrderNumber:6ES7 416-2FN05-0AB0	
OrderNumber:6ES7 455-0VS00-0AE0	V5.3
OrderNumber:6ES7 417-4XT07-0AB0	
OrderNumber:6ES7 421-1BL01-0AA0	V7.0
OrderNumber:6ES7 422-1BL00-0AA0	
OrderNumber:6ES7 452-1AH01-0AE0	V7.0
OrderNumber:6ES7 422-1BH11-0AA0	
OrderNumber:6ES7 421-7DH00-0AB0	
OrderNumber:6ES7 422-1FH00-0AA0	
OrderNumber:6ES7 422-1HH00-0AA0	

Whitelist	
TypIdentifier	FirmwareVersion
OrderNumber:6ES7 452-1AH00-0AE0	
OrderNumber:6ES7 455-1VS00-0AE0	
OrderNumber:6ES7 450-1AP01-0AE0	
OrderNumber:6GK7 443-5FX02-0XE0	
OrderNumber:6ES7 517-3FP00-0AB0	
OrderNumber:6ES7 515-2UM01-0AB0	
OrderNumber:6ES7 531-7KF00-0AB0	
OrderNumber:6ES7 211-1BE40-0XB0	V4.0
OrderNumber:6AG4114-2yxxx-xxxx//Device	V2.1
OrderNumber:6AG4112-2yxxx-xxxx//Device	V2.1
OrderNumber:6AG4114-2zxxx-xxxx//Device	V2.1
OrderNumber:6AG4132-2yxxx-xxxx//Device	V4.2
OrderNumber:6AV7884-0xyxx-xxxx//Device.12inch.Touch	
OrderNumber:6ES7647-7Byxx-xxxx//Device	
OrderNumber:6AV7884-5xyxx-xxxx//Device.19inch.Touch	
OrderNumber:6AV7884-2xyxx-xxxx//Device.15inch.Touch	
OrderNumber:6ES7647-7Byxx-xxxx//Device	
OrderNumber:6ES7647-7Bzxx-xxxx//Device	
OrderNumber:6AV7853-0xzxx-xxxx//Device.15inch.Touch	
OrderNumber:6AV7854-0xzxx-xxxx//Device.15inch.Key	
OrderNumber:6AV7854-0xxxx-xxxx//Device.15inch.Key	
OrderNumber:6ES7647-7Ayxx-xxxx//Device	
OrderNumber:6AV7856-0xyxx-xxxx//Device.19inch.Touch	
OrderNumber:6ES7647-7Ayxx-xxxx//Device	
OrderNumber:6AV7873-xxxxx-xBxx//Device.15inch.Key	
OrderNumber:6ES7643-8yxxx-xxxx//Device	
OrderNumber:6ES7647-6Byxx-xxxx//Device	
OrderNumber:6AV7870-xxxxx-xAxx//Device.12inch.Touch	
OrderNumber:6AV7872-xxxxx-xAxx//Device.15inch.Touch	
OrderNumber:6AV7874-xxxxx-xBxx//Device.17inch.Touch	
OrderNumber:6AV7875-xxxxx-xAxx//Device.19inch.Touch	
OrderNumber:6AG4112-0yxxx-xxxx//Device	
ET200eco - HeadModules	
ET200eco PN - HeadModules	
SIMATIC RF600 - HeadModules	
OrderNumber:3RK1 304-5KS40-2AA3	
OrderNumber:3RK1 304-5LS40-2AA3	
OrderNumber:3RK1 304-0HS00-8AA0	
OrderNumber:3RK1 304-0HS00-6AA0	
OrderNumber:6ES7 516-2PN00-0AB0	
OrderNumber:6ES7 151-3BB23-0AB0	
OrderNumber:6ES7 154-6AB00-0AB0	
OrderNumber:6ES7 141-5AH00-0BA0	
OrderNumber:6ES7 142-5AF00-0BA0	
OrderNumber:6ES7 143-5AH00-0BA0	

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6GK7 243-8RX30-0XE0	V3.0
OrderNumber:6ES7 212-1HE40-0XB0	V3.0
OrderNumber:6ES7 518-4FP00-0AB0	V1.0
OrderNumber:6AG1 215-1HG40-4XB0	V1.0
OrderNumber:6ES7 138-4FD00-0AA0	V2.1
OrderNumber:6ES7 134-4NB51-0AB0	V7.0
OrderNumber:6ES7 135-4GB52-0AB0	V1.0
OrderNumber:6ES7 138-7AA00-0AA0	V1.0
OrderNumber:6ES7 138-7BB00-0AB0	V1.0
OrderNumber:6ES7 145-4GF00-0AB0	V1.0
OrderNumber:6ES7 143-4BF50-0AA0	V3.0
OrderNumber:6GK5 980-3CB00-0AA1	V4.2
OrderNumber:6GK5 980-3CB00-0AA7	V2.1
OrderNumber:6GK5 991-4AB00-8AA0	V4.2
OrderNumber:6GK5 992-4AL00-8AA0	
OrderNumber:6GK5 992-4SA00-8AA0	
OrderNumber:6GK5 992-4RA00-8AA0	
OrderNumber:6GK5 992-4GA00-8AA0	
OrderNumber:6GK5 991-1AE00-8AA0	
OrderNumber:6GK5 991-1AF00-8AA0	
OrderNumber:6GK5 992-1AL00-8AA0	
OrderNumber:6GK5 992-1AN00-8AA0	
OrderNumber:6ES7 174-0AA10-0AA0	
OrderNumber:6GK5 204-0BA00-2BF2	
OrderNumber:6NH9 741-1AA00	
OrderNumber:6ES7 133-1BL01-0XB0	
OrderNumber:6ES7 132-1BL00-0XB0	
OrderNumber:6ES7 132-1BH00-0XB0	
OrderNumber:6ES7 131-1BL01-0XB0	
OrderNumber:6ES7 131-1BH01-0XB0	
OrderNumber:3RK1 207-2BQ44-0AA3 ET200eco - HeadModules	
ET200eco PN - HeadModules	
SIMATIC RF600 - HeadModules	
OrderNumber:3RK1 304-5KS40-2AA3	
OrderNumber:3RK1 304-5LS40-2AA3	
OrderNumber:3RK1 304-OHS00-8AA0	
OrderNumber:3RK1 304-OHS00-6AA0	
OrderNumber:6ES7 516-2PN00-0AB0	
OrderNumber:6ES7 151-3BB23-0AB0	
OrderNumber:6ES7 154-6AB00-0AB0	
OrderNumber:6ES7 141-5AH00-0BA0	
OrderNumber:6ES7 142-5AF00-0BA0	

Whitelist	
Typenidentifizier	FirmwareVersion
OrderNumber:6ES7 143-5AH00-0BA0	
OrderNumber:6GK7 243-8RX30-0XE0	
OrderNumber:6ES7 212-1HE40-0XB0	
OrderNumber:6ES7 518-4FP00-0AB0	
OrderNumber:6AG1 215-1HG40-4XB0	
OrderNumber:6ES7 138-4FD00-0AA0	
OrderNumber:6ES7 134-4NB51-0AB0	
OrderNumber:6ES7 135-4GB52-0AB0	
OrderNumber:6ES7 138-7AA00-0AA0	
OrderNumber:6ES7 138-7BB00-0AB0	
OrderNumber:6ES7 145-4GF00-0AB0	
OrderNumber:6ES7 143-4BF50-0AA0	
OrderNumber:6GK5 980-3CB00-0AA1	
OrderNumber:6GK5 980-3CB00-0AA7	
OrderNumber:6GK5 991-4AB00-8AA0	
OrderNumber:6GK5 992-4AL00-8AA0	
OrderNumber:6GK5 992-4SA00-8AA0	
OrderNumber:6GK5 992-4RA00-8AA0	
OrderNumber:6GK5 992-4GA00-8AA0	
OrderNumber:6GK5 991-1AE00-8AA0	
OrderNumber:6GK5 991-1AF00-8AA0	
OrderNumber:6GK5 992-1AL00-8AA0	
OrderNumber:6GK5 992-1AN00-8AA0	
OrderNumber:6ES7 174-0AA10-0AA0	
OrderNumber:6GK5 204-0BA00-2BF2	
OrderNumber:6NH9 741-1AA00	
OrderNumber:6ES7 133-1BL01-0XB0	
OrderNumber:6ES7 132-1BL00-0XB0	
OrderNumber:6ES7 132-1BH00-0XB0	
OrderNumber:6ES7 131-1BL01-0XB0	
OrderNumber:6ES7 131-1BH01-0XB0	
OrderNumber:3RK1 207-2BQ44-0AA3	
	V1.1
	V5.3
	V1.1

Die nachstehende Tabelle zeigt die Liste der Geräte/Geräteelemente, die den Import mit Hilfe von Bibliotheksverweisen nicht unterstützen.

Blacklist	
Typenidentifizier	FirmwareVersion
OrderNumber:6AG1 193-6AR00-7AA0	V0.0
OrderNumber:6ES7 193-6AR00-0AA0	V0.0
OrderNumber:6GK5 991-2VA00-8AA2	V0.0
OrderNumber:6ES7 193-6AM00-0AA0	V0.0
OrderNumber:6ES7 193-6AF00-0AA0	V0.0
OrderNumber:6ES7 193-6AP00-0AA0	V0.0
OrderNumber:6ES7 193-6AP20-0AA0	V0.0
OrderNumber:6ES7 193-6AP40-0AA0	V0.0
OrderNumber:6ES7 193-6AG00-0AA0	V0.0
OrderNumber:6ES7 193-6AG20-0AA0	V0.0
OrderNumber:6ES7 193-6AG40-0AA0	V0.0
OrderNumber:6ES7 193-6AP40-0AA0	V0.0
OrderNumber:6DL1 193-6AG00-0AA0	V0.0
OrderNumber:6DL1 193-6AF00-0AA0	V0.0
OrderNumber:6DL1 193-6AR00-0AA0	V0.0
OrderNumber:6DL1 193-6AG20-0AA0	V0.0
OrderNumber:6DL1 193-6AG40-0AA0	V0.0
OrderNumber:6ES7 655-5PX11-0XX0	V1.2
SIMATIC COMMUNICATION MODULES - HeadModules	
SIMATIC optical identification systems - HeadModules	
OrderNumber:6AV6 646-1AC22-0AX0	
OrderNumber:6GK5 826-2AB00-2AB2	
OrderNumber:6GK5 991-1AD00-8FA0	V2.0
SCALANCE XC208 - HeadModules	V5.0
SCALANCE XB208- HeadModules	
SETRON - HeadModules	
OrderNumber:6GK5 992-4AS00-8AA0	
Additional Ethernet devices	
Industrial PCs	
PCU 50.5	

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.24 Export eines Geräteelements

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist offline.

Verwendung

Das Objekt `Device` ist ein Behälterobjekt für eine zentrale oder dezentrale Konfiguration. Es ist das übergeordnete Objekt für `DeviceItem`-Objekte und das interne Element oberster Ebene der Instanzhierarchie des TIA Portal-Projekts innerhalb der Struktur einer AML-Datei.

Der CAX-Datenexport unterstützt folgende Arten von Geräten, angegeben über die AML-Typkennung:

- Physische Module
- GSD-/GSDML-basierte Geräte
- Systeme

Geräte können in ein `DeviceUserFolder` Objekt gruppiert werden.

Hinweis

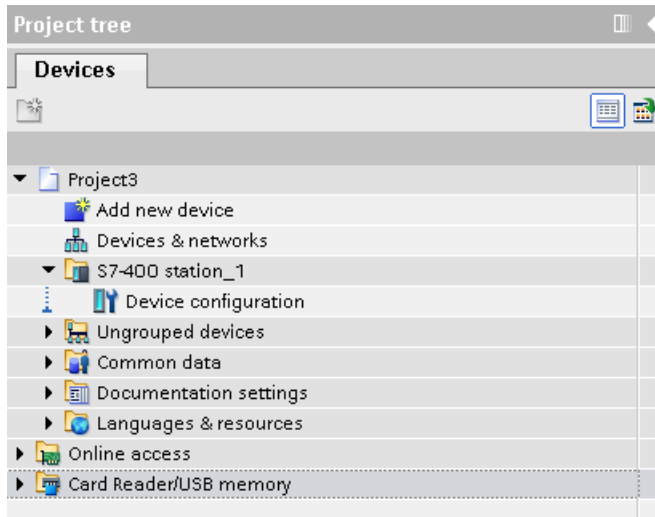
Beim Export eines Einzelgeräts werden auch alle Subnetze des Projekts exportiert.

Attribute

Die folgende Tabelle zeigt die zugehörigen Attribute des Geräteobjekts rechnergestützter Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	
Typidentifizier	Obligatorisch für Export	Optional für Import ab AR APC V1.1
Comment	Optional	Standardeinstellung: „“

Beispiel: Exportierte Konfiguration



AML-Struktur der Exportdatei

Das folgende Strukturbeispiel stellt den Export des Einzelgeräts "S7-400 station_1" ohne Baugruppenträger und Module dar:

```
...
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
  <InternalElement ID="288b7850-688e-43b3-941e-d615ba900a02" Name="Project3">
    <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
    <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
    <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
    <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
    <InternalElement ID="57611cfd-6da4-444e-ac78-5fbcea20a4e1" Name="S7-400 station_1">
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>System:Device.S7400</Value>
      </Attribute>
      <Attribute Name="Comment" AttributeDataType="xs:string">
        <Value>S7400 station</Value>
      </Attribute>
      <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/Device" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
  </InternalElement>
</InstanceHierarchy>
</CAEXFile>
```

Gerät ohne Typkennung und mit generischer Typkennung

Der CAx-Import muss in der Lage sein, Geräte ohne Typkennung oder mit generischen Typkennungen ("System:Device.Generic") zu verarbeiten. Es kann möglich sein, dass die AML-Datei Gerätetypen ohne Typkennung oder mit generischer Typkennung ("System:Device.Generic") enthält.

Aber mit dem CAx-Import können diese bearbeitet und Geräte korrekt erstellt werden.

Die folgenden Geräte unterstützen den Ersatz generischer Gerätekennungen oder fehlender Typkennungen:

- GSD- und GSDML-Geräte
- MDD-basierte Geräte (keine GSD/GSDML-Geräte)

Für generische Geräte oder Geräte ohne Typkennung und mit generischem Baugruppenträger und Ersatz der Typkennung muss das Kopfmodul (bei dezentralen Geräten) oder der PLC (bei zentralen Geräten) in dem in der AML-Datei beschriebenen Baugruppenträger vorhanden sein, sonst werden Geräte ohne Typkennungen oder der Ersatz der Typkennung für generische Geräte und generische Baugruppenträger nicht verarbeitet.

Die folgende XML-Struktur zeigt eine Gerätekonfiguration mit nicht-generischer Typkennung:

```
<InternalElement ID="04f5d5f08-316a-4a1d-9290-9bfd75b2b2ca" Name="S7-400 station_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Device.S7400</Value>
  </Attribute>
  <Attribute Name="Comment" AttributeDataType="xs:string">
    <Value>S7400 station</Value>
  </Attribute>
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Device"/>
</InternalElement>
```

Die folgende XML-Struktur zeigt eine Gerätekonfiguration mit generischer Typkennung:

```
<InternalElement ID="04f5d5f08-316a-4a1d-9290-9bfd75b2b2ca" Name="S7-400 station_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Device.Generic</Value>
  </Attribute>
  <Attribute Name="Comment" AttributeDataType="xs:string">
    <Value>S7400 Device</Value>
  </Attribute>
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Device"/>
</InternalElement>
```

Die folgende XML-Struktur zeigt eine Gerätekonfiguration mit Typkennung für das Gerät.

```
<InternalElement ID="a887601f-3ced-4f50-88ff-a9ec6eabb682" Name="S7-400 station_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Device.S7400</Value>
  </Attribute>
  <Attribute Name="Comment" AttributeDataType="xs:string">
    <Value>S7400 Device</Value>
  </Attribute>
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Device"/>
</InternalElement>
```

Die folgende XML-Struktur zeigt eine Gerätekonfiguration, in der keine Typkennung für das Gerät verfügbar ist.

```
<InternalElement ID="a887601f-3ced-4f50-88ff-a9ec6eabb682" Name="S7-400 station_1">
  <Attribute Name="Comment" AttributeDataType="xs:string">
    <Value>S7400 Device</Value>
  </Attribute>
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Device"/>
</InternalElement>
```

Siehe auch

Struktur der CAX-Daten zum Import/Export (Seite 1482)

AML-Typkennungen (Seite 1486)

6.5.25 Import eines Geräteobjekts

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Der PLC ist offline.

Verwendung

Das Objekt `Device` ist ein Behälterobjekt für eine zentrale oder dezentrale Konfiguration. Es ist das übergeordnete Objekt für `DeviceItem`-Objekte und das interne Element oberster Ebene der Instanzhierarchie des TIA Portal-Projekts innerhalb der Struktur einer AML-Datei.

Der CAX-Datenimport unterstützt folgende Arten von Geräten, angegeben über die AML-Typkennung:

- Physische Module
- GSD-/GSDML-basierte Geräte
- Systeme
- Generische Geräte

Wenn ein `DeviceUserFolder` Objekt bereits in einem TIA Portal-Projekt existiert, werden die Geräte in dem entsprechenden Ordner gruppiert.

Wenn Sie nur die Kennung (`TypeIdentifier`) eines Kopfmoduls oder eines PLC und nicht die des entsprechenden Baugruppenträgers und Geräts kennen, können Sie einen generischen Baugruppenträger importieren.

Beispiel: `TypeIdentifier = System:<Prefix>.Generic`

Für den Austausch generischer Geräte müssen die folgenden Elemente in dem in der AML Datei beschriebenen Baugruppenträger vorhanden sein.

- Zentrale Geräte: SPS
- Dezentrale Geräte: Kopfmodul

Sind die Geräte generisch, definiert das Attribut `BuiltIn` die Art des Baugruppenträgers oder Moduls:

- Physisch: `BuiltIn = True`
- Generisch: `BuiltIn = False`

Beispiel: Generische Geräte importieren

Das folgende Strukturbeispiel stellt den Import des generischen "S7-400 station"-Geräts ohne Baugruppenträger und Module dar.

```
...
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
<InternalElement ID="d4dc896a-f4a5-41b6-9c48-8d3a0a5a4343" Name="MyProject">
  <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
  <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
  <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
  <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
  <InternalElement ID="3e6277d1-1b12-4c18-b00e-25e3eac3ac35" Name="S7400 station_1">
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>System.Device.Generic</Value>
    </Attribute>
    <Attribute Name="Comment" AttributeDataType="xs:string">
      <Value>S7400 station_1</Value>
    </Attribute>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/Device" />
  </InternalElement>
  ...
  <SupportedRoleClass RefRoleClassPath=
    "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
</InternalElement>
</InstanceHierarchy>
</CAEXFile>
```

Beispiel: Importieren der Hierarchie eines Anwenderordners eines Geräts

Das folgende Strukturbeispiel zeigt den Import einer Ordnerhierarchie.

```
...
<InternalElement ID="4fe37f4f-2661-4492-95f0-3d8a8160c851" Name="Project1">
  <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
  <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
  <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
  <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
  <InternalElement ID="1ee1615f-9c67-432d-a7cc-b795babf67b6" Name="Group_1">
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
  </InternalElement>
  <InternalElement ID="ce14c85a-28de-41aa-ad08-2eb7d0fb755f" Name="Group_2">
    <InternalElement ID="852347e8-3c48-4eb9-8bd8-349d0c7caf34" Name="Group_3">
      <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
  </InternalElement>
  <InternalElement ID="97cf7924-1756-4e32-8716-ac18990e4762" Name="Group_4">
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
    "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
</InternalElement>
...
```

Importierte Anwenderordnerhierarchie

Der Name der Ordner für nicht gruppierte und nicht zugeordnete Geräte wird sprachspezifisch vergeben. Deshalb wird empfohlen, einen Import mit derselben Benutzeroberflächensprache wie beim Export durchzuführen. Andernfalls werden nicht gruppierte und nicht zugeordnete Geräte in Ordner importiert, die entsprechend der Sprache des Exports benannt sind.

Beispiel: Sie haben ein Projekt exportiert, das die Gerätesystemgruppe "Ungrouped devices" (englisch) enthält, und importieren die AML-Datei in einer deutschen Benutzeroberfläche. Das Ergebnis: Im Projekt wird eine Gerätesystemgruppe "Nicht gruppierte Geräte" (deutsch) angelegt, gleichzeitig wird beim CAX-Import eine Benutzergruppe "Ungrouped device" (englisch) erstellt.

Die folgende Hierarchie wird in die Projektnavigation importiert:

▼	Group_1	
▼	Group_2	
▼	Group_3	
▼	Group_4	

Siehe auch

Struktur der CAX-Daten zum Import/Export (Seite 1482)

AML-Typkennungen (Seite 1486)

6.5.26 Export/Import eines Geräts mit festgelegter Adresse**Voraussetzung**

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist offline.

Verwendung

Im TIA Portal können Sie die Adressobjekte von IO-Geräteelementen in eine AML-Datei exportieren. Beim Importieren in ein leeres TIA Portal-Projekt behalten die importierten Geräteelemente die Adressobjekte in den entsprechenden IO-Geräteelementen bei.

Das Attribut Address in der AML-Datei enthält RefSemantic mit der obligatorischen Einstellung auf den angegebenen Wert `OrderedListType`.

Bis TIA Portal V16 ist Voraussetzung für den Export und Import eines Geräteelements, dass die Adressen in der jeweils gleichen Reihenfolge stehen. Beispiel: Wenn ein Geräteelement im TIA-Portal zwei Adressen als Eingangs- und Ausgangsadresse unterstützt und dasselbe Geräteelement in der AML-Datei die Adressen in der Reihenfolge Ausgangs- und Eingangsadresse aufweist, werden die Adressen nicht verarbeitet, weil sie nicht in derselben Reihenfolge stehen.

Ab TIA Portal V17 wird bei einer CAX-Importoperation mit einer bestimmten Adresse die passende TIA-Adresse von der AML-Datei anhand des E/A-Typs ermittelt, und es wird versucht, die TIA-Adresse festzulegen. Die Reihenfolge in der AML-Datei spielt hier keine Rolle mehr.

Attribute eines Elements "Address"

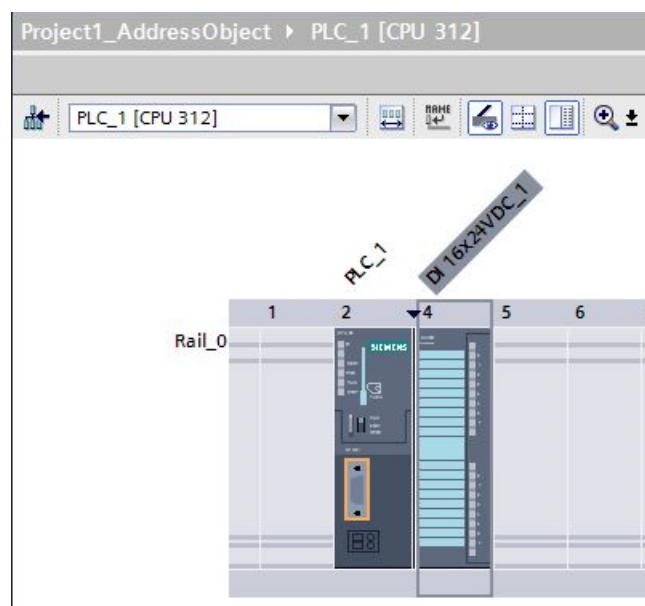
Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
IoType	Obligatorisch	Eingang oder Ausgang.
Length	Optional	Kanalbreite. Das Attribut wird beim Export immer in die Datei geschrieben
Start-Adresse	Optional	Anfangsadresse des IO-Geräts Das Attribut wird beim Export immer in die Datei geschrieben

Hinweis

- Die Funktion unterstützt einen beliebigen Geräteelement-Typ mit einem Adressobjekt vom Typ Eingang/Ausgang
- Eingangs- und Ausgangsadressen der Länge 0 werden vom Export ausgeschlossen und beim Import ignoriert
- Bei Adressen mit der Startadresse -1 wird der Export des Adressattributs übersprungen. Beim Import wird für Adressen mit der Startadresse -1 jedoch eine Warnmeldung angezeigt

Beispiel: IO-Geräteelemente mit Adressobjekten



AML-Struktur

Die folgende Abbildung zeigt eine partielle Elementstruktur der exportierten AML-Datei. Sie enthält die Address-Elemente und deren Attribute.

```
<Attribute Name="Address">
<RefSemantic CorrespondingAttributePath="OrderedListType" />
<Attribute Name="1">
<Attribute Name="StartAddress" AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>16</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
</Attribute>
</Attribute>
```

Pruned XML

Pruning beschreibt den Vorgang der Optimierung des Inhalts durch Entfernen bestimmter Dinge, die in der XML nicht unbedingt angegeben werden müssen. In dieser reduzierten XML werden die automatisch erstellten Submodulinformationen nicht angegeben, und die entsprechenden Adressobjekte werden unter dem direkt übergeordneten Element angeordnet.

Die folgende Abbildung zeigt eine partielle Elementstruktur der exportierten AML-Datei vor dem Pruning.

```
<InternalElement ID="5511a117-42c6-44b7-be5d-0f33cd46e932" Name="AS-i Master_1">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>4</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>True</Value>
</Attribute>
<Attribute Name="Address">
<RefSemantic CorrespondingAttributePath="OrderedListType"/>
<Attribute Name="1">
<Attribute Name="StartAddress" AttributeDataType="xs:int">
<Value>20</Value>
</Attribute>
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>256</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
</Attribute>
```

In der Pruned-AML-Datei werden die Submodulinformationen like <InternalElement> entfernt und die entsprechenden Adressobjekte beibehalten.

```
<Attribute Name="Address">
<RefSemantic CorrespondingAttributePath="OrderedListType"/>
<Attribute Name="1">
<Attribute Name="StartAddress" AttributeDataType="xs:int">
<Value>20</Value>
</Attribute>
<Attribute Name="Length" AttributeDataType="xs:int">
<Value>256</Value>
</Attribute>
<Attribute Name="IoType" AttributeDataType="xs:string">
<Value>Input</Value>
</Attribute>
</Attribute>
</Attribute>
```

Siehe auch

Pruned AML (Seite 1478)

6.5.27 Export/Import eines Geräts mit Kanälen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist offline.

Verwendung

Im TIA Portal können Sie die Kanalobjekte von IO-Geräteelementen in eine AML-Datei exportieren. Beim Importieren in ein leeres TIA Portal-Projekt behalten die importierten Geräteelemente die Kanalobjekte in den entsprechenden IO-Geräteelementen bei.

Das Element <ExternalInterface> wird in internen Elementen von Knoten und Subnetzen dargestellt und gibt an, dass Knoten und Subnetze verbunden sind.

Attribute eines Elements "ExternalInterface"

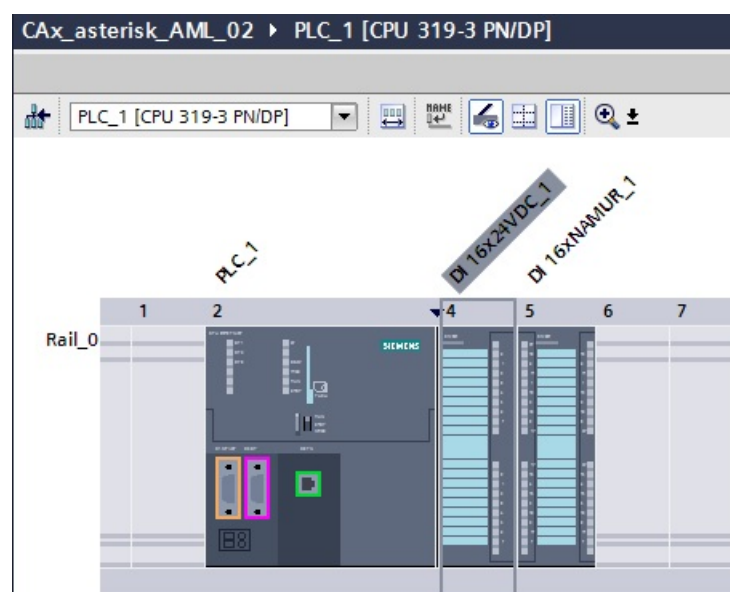
Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
IoType	Obligatorisch	Eingang oder Ausgang
Length	Optional	Kanalbreite (1 für digitale und 16 für analoge Signale)
Number	Obligatorisch	Kanalnummer beginnt bei 0
Type	Obligatorisch	Analog oder digital

Kanalnummerierung

Digitaleingangs-, Digitalausgangs-, Analogeingangs-, Analogausgangs- und Technologiekanäle werden als DI_0, DO_0, AI_0, AO_0, TO_0 nummeriert. Die Kanäle auf den Geräteelementen selbst werden zuerst nummeriert, und die Kanäle auf Untergeräteelementen werden nachfolgend nummeriert (Tiefe zuerst). Jedes weitere Geräteelement hat eine eigene Kanalnummer, beginnend bei 0.

Beispiel: Geräte mit Kanälen



AML-Struktur

Die folgende Abbildung zeigt eine partielle Elementstruktur der exportierten AML-Datei.

```
<ExternalInterface ID="31ca16d3-6322-43b6-95bc-e2d7d7bfc7b7" Name="Channel_DI_0"
  RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
  <Attribute Name="Type" AttributeDataType="xs:string">
    <Value>Digital</Value>
  </Attribute>
  <Attribute Name="IoType" AttributeDataType="xs:string">
    <Value>Input</Value>
  </Attribute>
  <Attribute Name="Number" AttributeDataType="xs:int">
    <Value>0</Value>
  </Attribute>
  <Attribute Name="Length" AttributeDataType="xs:int">
    <Value>1</Value>
  </Attribute>
</ExternalInterface>
```

6.5.28 Export von Geräteelementobjekten

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist offline.

Verwendung

Der Export von Geräteelementobjekten ist nur auf PLC-Geräte anwendbar.

`DeviceItem`-Objekte sind verschachtelte untergeordnete Elemente eines `Device`-Objekts:
Ein Objekt vom Typ `DeviceItem` kann ein Baugruppenträger oder ein Einschubmodul sein.

- Das erste Nachfolgeelement eines Geräts ist vom Typ "Baugruppenträger".
Die `PositionNumber` eines Baugruppenträgers beginnt bei 0. Gibt es mehrere Baugruppenträger, werden diese fortlaufend nummeriert (1, 2, 3, ...).
Es gibt keine Einschränkungen bezüglich der Reihenfolge innerhalb eines Hierarchielevels in der AML Datei.
- Alle weiteren Nachfolger des Typs "Baugruppenträger" sind Module.

Der CAX-Datenexport unterstützt folgende Arten von Geräteelemente, angegeben über die AML-Typkennung:

- Physische Module
- GSD/GSDML-Module

Attribute

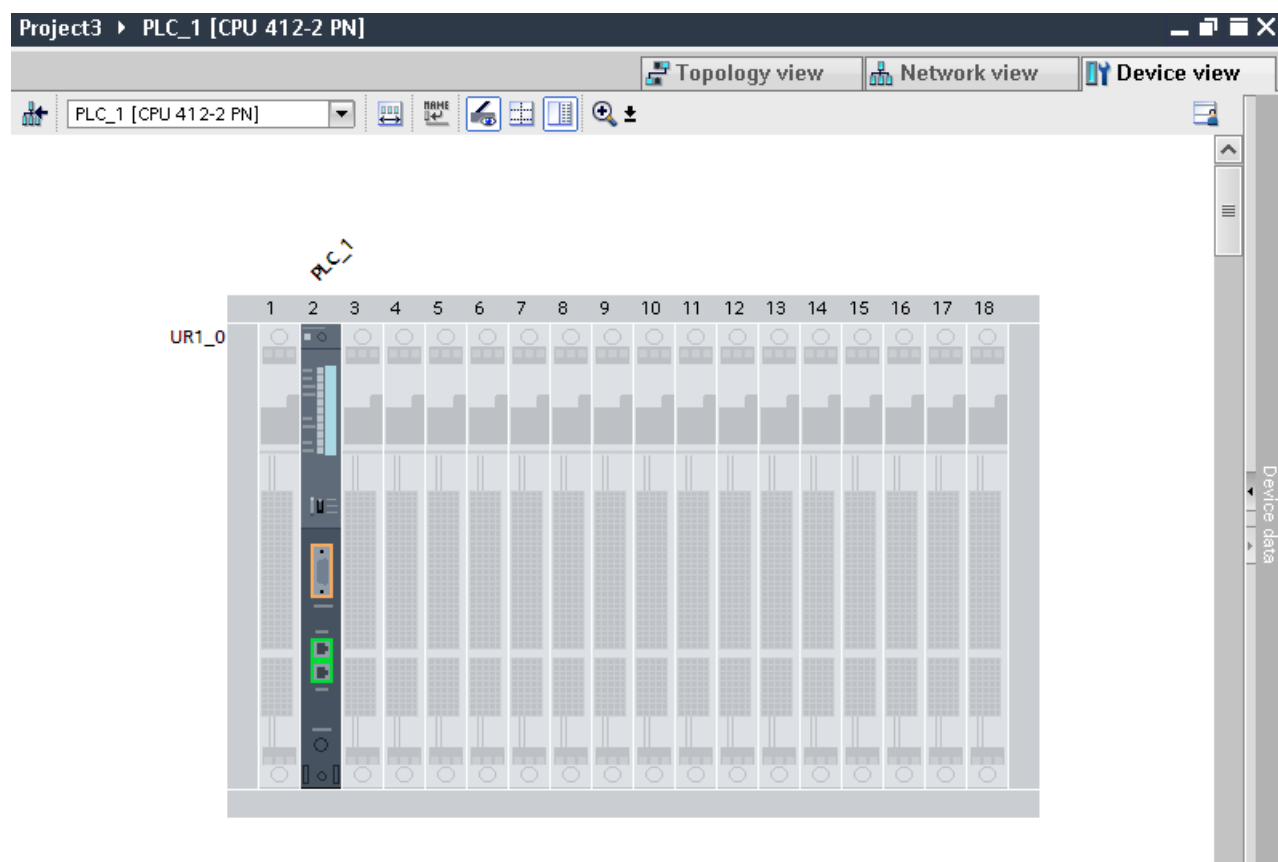
Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch Nur Exportieren für "BuiltIn" = TRUE	
TypeName	Nur Exportieren für "BuiltIn" = FALSE	
DeviceItemtype	Nur Exportieren	Nur für PLCs (Zentralgeräte) und Geräteelemente (physische Baugruppenträger, Module, Kopfmodul). Optional beim Import, doch jedes Geräteelement außer Basiseinheiten mit DeviceItemtype als Zubehör wird stillschweigend ignoriert.
PositionNumber	Obligatorisch Optional für bestimmte Geräteelemente	Für bestimmte Geräteelemente ist die Positionsnummer für den Austausch mit externen Tools wie ECAD-Systemen nicht relevant. Für solche Geräteelemente ist die Positionsnummer in der AML-Datei nicht obligatorisch und wird intern beim Exportieren mit einem entsprechenden Wert belegt.
BuiltIn	Optional	Standardeinstellung: FALSE
TypeIdentifier	Obligatorisch für "BuiltIn" = FALSE Ignoriert für "BuiltIn" = TRUE	Bei integrierten Geräteelementen wird dieses Attribut mit der Typkennung seines steckbaren übergeordneten Elements exportiert. Es hat keine Relevanz beim Import und ist somit optional. Bei nicht integrierten Geräteelementen ist dieses Attribut optional.
FirmwareVersion	Optional, obligatorisch wenn das Objekt Firmware-Versionen unterstützt	
PlantDesignation IEC	Optional	Standardeinstellung: ""
LocationIdentifier IEC	Optional	Standardeinstellung: ""
Comment	Optional für "BuiltIn" = FALSE	Standardeinstellung: ""
ProductDesignation IEC	Obligatorisch wenn das Objekt dieses Attribut in TIA Portal unterstützt und nicht leer ist	Das Importieren wird nicht unterstützt, wenn die Länge des Product-Designation IEC größer als 54 Zeichen ist. Export/Import unter Geräteelementen als Teil des AR APC V1.1.0 Supports für TIA Portal V16
InstallationDate	Obligatorisch wenn das Objekt dieses Attribut in TIA Portal unterstützt	Export/Import unter Geräteelementen als Teil des AR APC V1.1.0-Supports für TIA Portal V16
Address	Optional	"Address" besitzt verschachtelte Attribute

Die folgende Tabelle zeigt die verschachtelten Attribute der „Address“ Attribute des Objekts „Devicetern“:

Attribute für „Address“	Handha- bung	Kommentar
StartAddress	Obligato- risch	
Length	Nur Expor- tieren	Der Export/Import von Adressen mit der Länge = 0 wird nicht unterstützt.
IoType	Obligato- risch	Eingang oder Ausgang

Beispiel: Exportierte Konfiguration



AML-Struktur der Exportdatei

Das folgende Strukturbeispiel stellt den Export von "UR1_0" und dem Modul "PLC_1" dar.

6.5 Hardware-Daten importieren/exportieren

```

<InternalElement ID="7624ed42-3db7-4ba5-8726-e719d7b09969" Name="S7-400 station_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Device.S7400</Value>
  </Attribute>
  <Attribute Name="Comment" AttributeDataType="xs:string">
    <Value>S7400 station</Value>
  </Attribute>
</InternalElement ID="1ff247e6-6b94-42a2-bcd5-d673fc6e9ba5" Name="UR1_0">
  <Attribute Name="TypeName" AttributeDataType="xs:string">
    <Value>UR1</Value>
  </Attribute>
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>0</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>False</Value>
  </Attribute>
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>OrderNumber:6ES7 400-1TA01-0AA0</Value>
  </Attribute>
  <Attribute Name="Comment" AttributeDataType="xs:string">
    <Value>S7 400 rack</Value>
  </Attribute>
<InternalElement ID="202421bf-a4b9-4399-a563-9f154f0eabab" Name="PLC_1">
  <Attribute Name="TypeName" AttributeDataType="xs:string">
    <Value>CPU 412-2 PN</Value>
  </Attribute>
  <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
    <Value>CPU</Value>
  </Attribute>
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>2</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>False</Value>
  </Attribute>
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>OrderNumber:6ES7 412-2EK06-0AB0</Value>
  </Attribute>
  <Attribute Name="Comment" AttributeDataType="xs:string">
    <Value>S7 400 plc</Value>
  </Attribute>
  <Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
    <Value>V6.0</Value>
  </Attribute>
  <Attribute Name="ProductDesignation IEC" AttributeDataType="xs:string">
    <Value>Additional Information</Value>
  </Attribute>
  <Attribute Name="InstallationDate" AttributeDataType="xs:dateTime">
    <Value>2018-11-26T05:09:02.803Z</Value>
  </Attribute>
  <Attribute Name="PlantDesignation IEC" AttributeDataType="xs:string">
    <Value>PD</Value>
  </Attribute>
  <Attribute Name="LocationIdentifier IEC" AttributeDataType="xs:string">
    <Value>LI</Value>
  </Attribute>
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Device" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/AutomationProject" />
</InternalElement>

```

Siehe auch

Export/Import von GSD/GSDML-basierten Geräten und Geräteelementen (Seite 1626)

Struktur der CAx-Daten zum Import/Export (Seite 1482)

AML-Typkennungen (Seite 1486)

6.5.29 Import von Geräteelementobjekten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal aufbauen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)
- Die PLC ist offline.

Verwendung

Der Import von Geräteelementobjekten ist nur auf PLC-Geräte anwendbar.

`DeviceItem`-Objekte sind verschachtelte untergeordnete Elemente eines `Device`-Objekts: Ein Objekt vom Typ `DeviceItem` kann ein Baugruppenträger oder ein Einschubmodul sein.

- Das erste Nachfolgeelement eines Geräts ist vom Typ Baugruppenträger.
Die `PositionNumber` eines Baugruppenträgers beginnt bei 0. Gibt es mehrere Baugruppenträger, werden diese fortlaufend nummeriert (1, 2, 3, ...).
Es gibt keine Einschränkungen bezüglich der Reihenfolge innerhalb eines Hierarchielevels in der AML Datei.
- Alle weiteren Nachfolger des Typs Baugruppenträger sind Module.

Der CAx-Datenimport unterstützt folgende Arten von Geräteelemente, angegeben über die AML-Typkennung:

- Physische Module
- GSD/GSDML-Module
- Generische Module

Wenn Sie nur die Kennung (`TypeIdentifier`) eines Kopfmoduls oder eines PLC und nicht die des entsprechenden Baugruppenträgers und Geräts kennen, können Sie einen generischen Baugruppenträger importieren.

Beispiel: `TypeIdentifier = System:Rack.Generic`

Für den Austausch generischer Baugruppenträger müssen die folgenden Elemente in dem in der AML Datei beschriebenen Baugruppenträger vorhanden sein:

- Zentrale Geräte: SPS
- Dezentrale Geräte: Kopfmodul

Ein generischer Baugruppenträger stammt aus dem Device-Typ. Demnach kann eine AML-Datei, die in das TIA Portal importiert wird, die Typkennung dieses Baugruppenträgers verwenden:

In diesem Fall bestimmt das TIA Portal die Typkennung für den Baugruppenträger.

Sind Baugruppenträger und Module generisch, definiert das Attribut `BuiltIn` die Art des Baugruppenträgers oder Moduls:

- Physisch: `BuiltIn = True`
- Generisch: `BuiltIn = False`

Einschränkungen

Während des Imports ist das Attribut `DeviceItemType` nicht relevant und ist optional.

Hinweis

Attribut "FirmwareVersion"

Wenn in der Importdatei keine `FirmwareVersion` angegeben ist, wird beim CAx-Import die letzte Firmwareversion verwendet, die im TIA Portal verfügbar ist.

Wenn das Attribut `FirmwareVersion` in der Importdatei mit einem leeren Wert vorhanden ist, schlägt der Import des Geräteelements fehl und es wird eine Fehlermeldung protokolliert.

Beispiel: Generische Geräte importieren

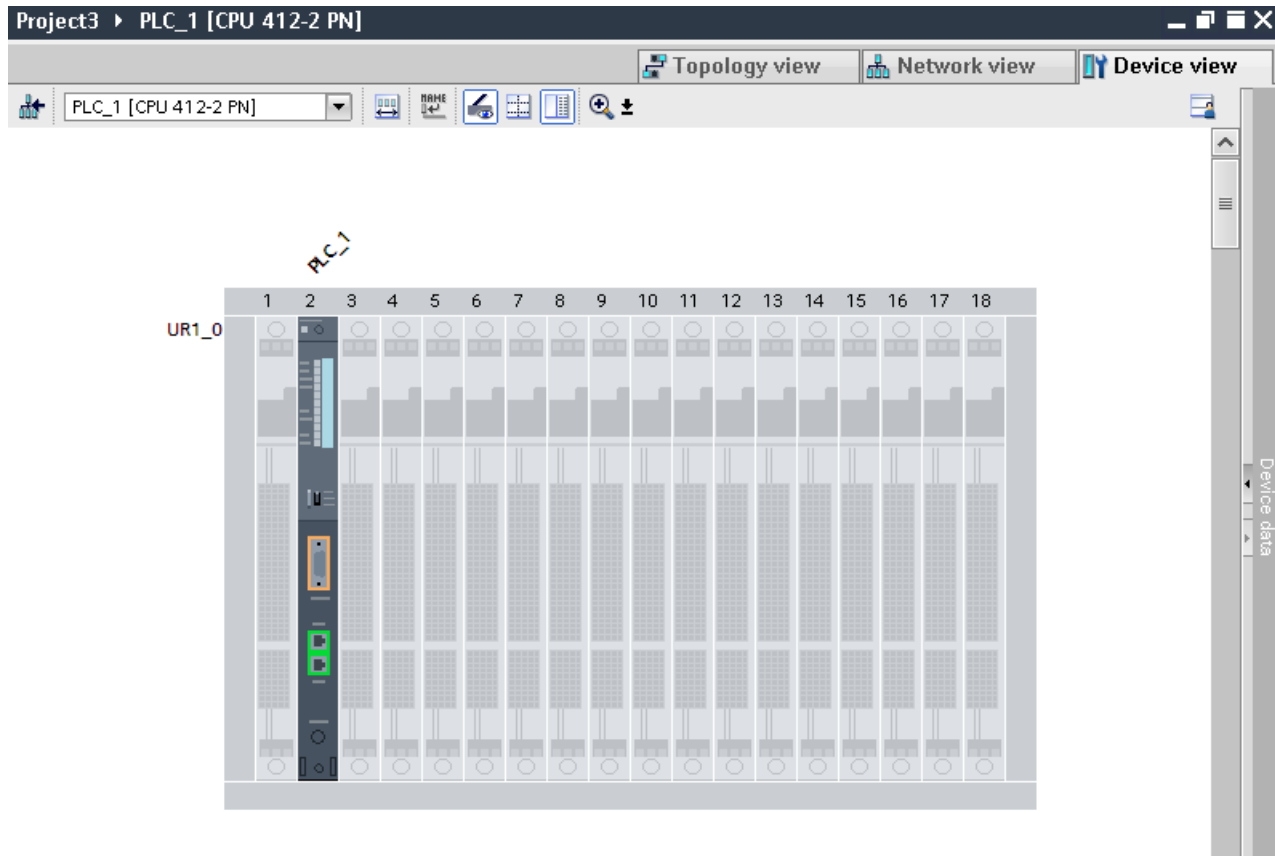
Das folgende Strukturbeispiel stellt den Import des generischen Baugruppenträgers „Rack_1“ dar.

```
...
<InternalElement ID="6563466e-2de9-42ca-951d-eb8f2545958d" Name="S7-400_station_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Device.S7400</Value>
  </Attribute>
</InternalElement>
<InternalElement ID="96930368-14ec-43e2-b9b7-c1fefc4b0534" Name="UR1_0">
  <Attribute Name="TypeName" AttributeDataType="xs:string">
    <Value>UR1</Value>
  </Attribute>
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>0</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>False</Value>
  </Attribute>
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Rack.Generic</Value>
  </Attribute>
</InternalElement>
<InternalElement ID="alde449e-4f89-45af-8bbc-f77c28bccd04" Name="PLC_1">
  <Attribute Name="TypeName" AttributeDataType="xs:string">
    <Value>CPU 412-2 PN</Value>
  </Attribute>
  <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
    <Value>CPU</Value>
  </Attribute>
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>2</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>False</Value>
  </Attribute>
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>OrderNumber:6ES7 412-2EK06-0AB0</Value>
  </Attribute>
  <Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
    <Value>V6.0</Value>
  </Attribute>

  <SupportedRoleClass RefRoleClassPath=
    "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath=
  "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath=
  "AutomationProjectConfigurationRoleClassLib/Device" />
</InternalElement>
...
```

Importierte Konfiguration

Die folgende Abbildung zeigt die importierte Konfiguration in der TIA Portal-Nutzerschnittstelle:



Siehe auch

Struktur der CAx-Daten zum Import/Export (Seite 1482)

AML-Typkennungen (Seite 1486)

6.5.30 Export/Import von GSD/GSDML-basierten Geräten und Geräteelementen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist offline.

Verwendung

Der CAX-Import/Export von GSD/GSDML-basierten Geräten und Geräteelementen ist mit dem Import/Export von Standardgeräten vergleichbar.

Bei GSD/GSDML-basierten Geräten und Geräteelementen weichen die exportierbaren Attribute voneinander ab, z. B. bei GSD/GSDML ist das Attribut "Label" vorhanden.

Generischer Import von Geräten und Baugruppenträgern ist möglich. Verwenden Sie für den Import die gleiche Kennung wie für Standardgeräte:

- Generische Geräte importieren: `TypeIdentifier = System:Device.Generic`
- Generische Baugruppenträger importieren: `TypeIdentifier = System:Rack.Generic`

Sind die Geräte generisch, definiert das Attribut `BuiltIn` die Art:

- Physisch: `BuiltIn = True`
- Generisch: `BuiltIn = False`

Attribute für ein Gerät

Die folgende Tabelle zeigt die zugehörigen Attribute von Geräten von CAX-Import- und -Exportdateien:

Attribut	Handhabung von Attribut	Kommentar
Name	Obligatorisch für Export und Import	
TypeIdentifier	Obligatorisch für Export und optional für Import ab AR APC V1.1	
Comment	Optional für Import	

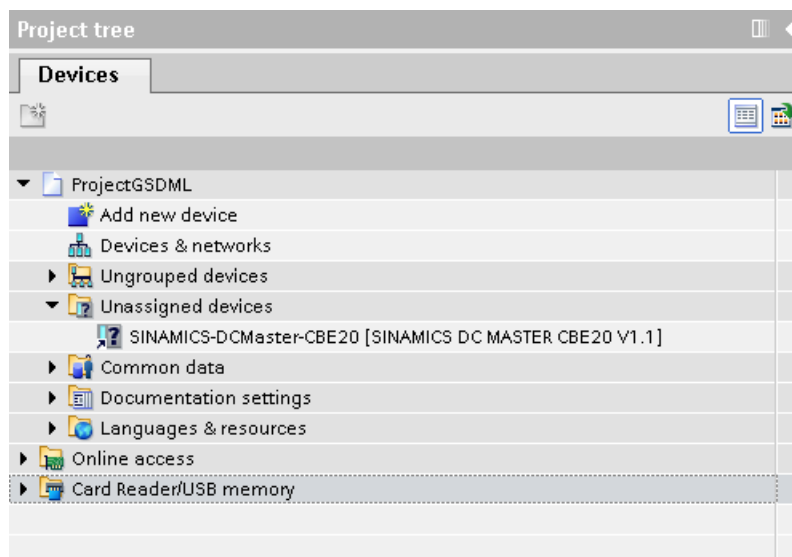
Attribute für ein Geräteelement

Die folgende Tabelle zeigt die zugehörigen Attribute eines Geräteelements von CAX-Import- und -Exportdateien:

Attribut	Handhabung von Attribut Integriert = FALSCH Generische Geräteelemente	Handhabung von Attribut Integriert = WAHR Physische Geräteelemente	Kommentar
Name	Obligatorisch	Nur Exportieren	
TypeName	Nur Exportieren	-/-	
DeviceItem-Type	Nur Exportieren	Nur Exportieren	Nur für PLC-Geräteelemente (zentrale Geräte) und Kopfmodul-Geräteelemente (dezentrale Geräte) Optional beim Import, doch jedes Geräteelement außer Basiseinheiten mit DeviceItem-Type als Zubehör wird ignoriert.

Attribut	Handhabung von Attribut Integriert = FALSCH Generische Geräteelemente	Handhabung von Attribut Integriert = WAHR Physische Geräteelemente	Kommentar
PositionNum- ber	Obligatorisch	Obligatorisch für Export Ausnahmefälle: Geräteelemente vom Typ Schnittstelle: Optional für Import Geräteelemente vom Typ Port: Optional	
BuiltIn	Optional		Standardeinstellung: FALSE
Typenidentifizier	Obligatorisch für „BuiltIn“ = FALSE	Ignoriert für „BuiltIn“ = TRUE	Bei integrierten Geräteelementen wird dieses Attribut mit der Typkennung seines steckbaren übergeordneten Elements exportiert. Das Attribut hat beim Import keine Relevanz und ist somit optional. Bei nicht integrierten Geräteelementen hat dieses Attribut keine Relevanz.
Comment	Optional	-	
Label	-	- Geräteelemente vom Typ Schnittstelle: Obligatorisch Geräteelemente vom Typ Port: Obligatorisch	

Beispiel: Exportiertes GSD-/GSDML-Gerät



AML-Struktur der Exportdatei

Die folgende Abbildung zeigt die Struktur der exportierten AML-Datei.

```

...
<InternalElement ID="9ae02cde-dfb4-4d43-a649-68b9ede7fc3d" Name="Ungrouped devices">
  <InternalElement ID="12d4ce0f-346d-4bfa-b139-c9d0db64c794" Name="GSD_device_1">
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCMaster-20140704.XML/D</Value>
    </Attribute>
    <InternalElement ID="ccb1cb62-67b2-4b8c-951f-10c7ffb4d787" Name="Rack">
      <Attribute Name="TypeName" AttributeDataType="xs:string">
        <Value>Rack</Value>
      </Attribute>
      ...
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCMaster-20140704.XML/R/IDD_14</Value>
      </Attribute>
      <InternalElement ID="74f25b5c-0c09-46d0-9011-f341a3e98a0d" Name="SINAMICS-DCMaster-CBE20">
        <Attribute Name="TypeName" AttributeDataType="xs:string">
          <Value>SINAMICS DC MASTER CBE20 V1.1</Value>
        </Attribute>
        <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
          <Value>HeadModule</Value>
        </Attribute>
        <Attribute Name="PositionNumber" AttributeDataType="xs:int">
          <Value>0</Value>
        </Attribute>
        <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
          <Value>False</Value>
        </Attribute>
        <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
          <Value>GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCMaster-20140704.XML/DAP/IDD_14</Value>
        </Attribute>
        <InternalElement ID="94f34bb9-fe47-4904-b8a1-62e8fb6b1b74"
          Name="SINAMICS DC MASTER CBE20 V1.1">
          <Attribute Name="PositionNumber" AttributeDataType="xs:int">
            <Value>0</Value>
          </Attribute>
          <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
            <Value>True</Value>
          </Attribute>
          <SupportedRoleClass RefRoleClassPath=
            "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
        </InternalElement>
        ...
        <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
      </InternalElement>
      <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/Device" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
    "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
</InternalElement>
...

```

Generischer und nicht-generischer Baugruppenträger mit und ohne Typidentifizier

Der CAX-Import muss in der Lage sein, Geräte ohne Typkennung oder mit generischen Typkennungen, d. h. 'System:Device.Generic', und Baugruppenträger mit generischen Typkennungen, d. h. 'System:Rack.Generic', zu verarbeiten.

Während des Importierens kann es möglich sein, dass die AML-Datei Gerätetypen ohne Typkennung oder mit generischer Typkennung, d. h. 'System:Device.Generic', und Baugruppenträger-Geräteelemente mit generischer Typkennung, d. h. 'System:Rack.Generic', enthält. Aber mit dem CAX-Import können diese bearbeitet sowie Geräte und Baugruppenträger-Geräteelemente korrekt erstellt werden.

Die folgenden Geräte unterstützen den Ersatz generischer Geräte, von Geräten ohne Typkennung und generischer Baugruppenträger:

- GSD- und GSDML-Geräte - Alle GSD- und GSDML-Geräte mit GSD/GSDML-Baugruppenträgern.
- MDD-basierte Geräte (keine GSD/GSDML-Geräte) - Geräte mit Systembaugruppenträger-Typkennungen.

Für die Verarbeitung generischer Baugruppenträger ist der CAX-Export nicht relevant. CAX exportiert immer Typkennungen für nicht-generische Baugruppenträger.

Für generische Geräte oder Geräte ohne Typkennung und mit generischem Baugruppenträger und Ersatz der Typkennung muss das Kopfmodul (bei dezentralen Geräten) oder der PLC (bei zentralen Geräten) in dem in der AML-Datei beschriebenen Baugruppenträger vorhanden sein, sonst werden Geräte ohne Typkennungen oder der Ersatz der Typkennung für generische Geräte und generische Baugruppenträger nicht verarbeitet.

Die folgende XML-Struktur zeigt eine GSDML-Gerätekonfiguration mit Gerät (mit Typkennung) und Baugruppenträger (mit Typkennung):

```
<InternalElement ID="bc3b50fa-5cfa-4bf3-a496-8e46080d4f86" Name="GSD device_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>GSD:GSDML-V2.32-SIEMENS-SINAMICS_DCMaster-20160531.XML/D</Value>
  </Attribute>
  <InternalElement ID="c80f2d97-66c9-4f31-bf6b-17e3e0de0509" Name="Rack">
    <Attribute Name="TypeName" AttributeDataType="xs:string">
      <Value>Rack</Value>
    </Attribute>
    <Attribute Name="PositionNumber" AttributeDataType="xs:int">
      <Value>0</Value>
    </Attribute>
    <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
      <Value>false</Value>
    </Attribute>
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>GSD:GSDML-V2.32-SIEMENS-SINAMICS_DCMaster-20160531.XML/R/IDD_14</Value>
    </Attribute>
    <InternalElement ID="f519b4b9-b1f7-4011-bed2-25be85c8c2a8" Name="SINAMICS-DCMaster-CBE20">
      <Attribute Name="TypeName" AttributeDataType="xs:string">
        <Value>SINAMICS DC MASTER CBE20 V1.1</Value>
      </Attribute>
      <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
        <Value>HeadModule</Value>
      </Attribute>
```

Die folgende XML-Struktur zeigt eine GSDML-Gerätekonfiguration mit Gerät (ohne Typkennung) und Baugruppenträger (generische Typkennung):

```
<InternalElement ID="bc3b50fa-5cfa-4bf3-a496-8e46080d4f86" Name="GSD device_1">
<InternalElement ID="c80f2d97-66c9-4f31-bf6b-17e3e0de0509" Name="Rack">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>Rack</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>0</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>System:Rack.Generic</Value>
</Attribute>
<InternalElement ID="f519b4b9-b1f7-4011-bed2-25be85c8c2a8" Name="SINAMICS-DCMaster-CBE20">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>SINAMICS DC MASTER CBE20 V1.1</Value>
</Attribute>
<Attribute Name="DeviceItemType" AttributeDataType="xs:string">
<Value>HeadModule</Value>
</Attribute>
```

Die folgende XML-Struktur zeigt eine GSDML-Gerätekonfiguration mit Gerät (generische Typkennung) und Baugruppenträger (generische Typkennung):

```
<InternalElement ID="bc3b50fa-5cfa-4bf3-a496-8e46080d4f86" Name="GSD device_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Device.Generic</Value>
  </Attribute>
  <InternalElement ID="c80f2d97-66c9-4f31-bf6b-17e3e0de0509" Name="Rack">
    <Attribute Name="TypeName" AttributeDataType="xs:string">
      <Value>Rack</Value>
    </Attribute>
    <Attribute Name="PositionNumber" AttributeDataType="xs:int">
      <Value>0</Value>
    </Attribute>
    <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
      <Value>false</Value>
    </Attribute>
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>System:Rack.Generic</Value>
    </Attribute>
    <InternalElement ID="f519b4b9-b1f7-4011-bed2-25be85c8c2a8" Name="SINAMICS-DCMaster-CBE20">
      <Attribute Name="TypeName" AttributeDataType="xs:string">
        <Value>SINAMICS DC MASTER CBE20 V1.1</Value>
      </Attribute>
      <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
        <Value>HeadModule</Value>
      </Attribute>
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>GSD:GSDML-V2.32-SIEMENS-SINAMICS_DCMaster-20160531.XML/DAP/IDD_14</Value>
      </Attribute>
    </InternalElement>
  </InternalElement>
</InternalElement>
```

Siehe auch

[AML-Typkennungen \(Seite 1486\)](#)

[Verbindung zum TIA Portal aufbauen \(Seite 84\)](#)

[Projekt öffnen \(Seite 131\)](#)

6.5.31 Exportieren/Importieren der Gerätekonfiguration mit virtueller Schnittstelle

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist offline.

Anwendung

Im TIA Portal werden Ports für die Kommunikation zwischen den Geräten verwendet und sind unter den Schnittstellen zu finden. Es gibt jedoch bestimmte Arten von Geräten, bei denen sich die Ports direkt unter den Geräteelementen befinden, bei denen es sich nicht um Schnittstellen handelt. Das entspricht nicht dem AML-Standard, der vorgibt, dass Ports immer unter den Schnittstellen zu finden sind.

CAX verwendet eine imaginäre Schnittstelle, die als virtuelle Schnittstelle bezeichnet wird, um die Gerätekonfiguration zu exportieren und zu importieren, wenn sich die Ports direkt unter Geräteelementen befinden, die keine Schnittstellen sind.

Export einer AML-Datei

Das folgende Beispiel zeigt eine AML-Datei mit virtueller Schnittstelle:

```
<InternalElement ID="822622a0-056a-494c-a802-2463c5e1b47d" Name="SCALANCE interface_1">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<InternalElement ID="5a604a57-bc2d-4763-8df0-7d20000faf1b" Name="VirtualInterface_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Ethernet</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<InternalElement ID="3c4862a1-b8ed-4610-88f3-29dc8328e748" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="220dd49d-8d23-44b1-bdc1-878516540313" Name="Port_2">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P2</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="99c6253b-c546-4720-af54-92db926b8231"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
```

Die virtuelle Schnittstelle wird bis TIA Portal V16 mit den folgenden Attributen exportiert:

- Name: ScalancelInterface_1
- Bezeichnung: Switch

Ab TIA Portal V16 wird die virtuelle Schnittstelle mit den folgenden Attributen exportiert:

- Name: VirtualInterface_1
- Bezeichnung: X1
- Typ: Ethernet

Importieren

CAX unterstützt den Import von virtuellen Schnittstellen. Dann werden die Ports unter der virtuellen Schnittstelle in der AML-Datei verarbeitet, unter dem richtigen übergeordneten Element im TIA Portal. Hier wird jede Schnittstelle mit der Bezeichnung "Switch" als virtuelle Schnittstelle betrachtet. Ab TIA Portal V16 ist die Bezeichnung jedoch keine Kennung mehr, da sie in X1 geändert wurde, was einer echten Schnittstelle entspricht.

Für das Typattribut wurde für die virtuelle Schnittstelle trotzdem "Ethernet" festgelegt, es ist optional. Ab TIA Portal V16 werden also alle Schnittstellenelemente in der AML-Datei, die nicht als "Switch" bezeichnet sind, auf allgemeine Weise behandelt, und CAX versucht bei jedem Vorkommen in der AML-Datei den Abruf aus dem TIA Portal.

Bei virtuellen Schnittstellen misslingt CAX die Suche, doch die darin enthaltenen Ports werden verarbeitet. Findet CAX jedoch eine echte Schnittstelle nicht, werden die darin enthaltenen Ports nicht verarbeitet.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.32 Exportieren/Importieren von Subnetzen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist offline.

AML-Struktur

Subnetze beschreiben ein physisches Netzwerk, insbesondere, welche Geräte an dem gleichen Netzwerk vom Typ PROFIBUS, PROFINET, MPI oder ASI angeschlossen sind.

Die Verbindungen zwischen einem Netzwerk und den Geräteelementen werden als Referenz für das Netzwerkobjekt abgebildet. Es gibt keine Referenz von den Netzwerkobjekten zu den verbundenen Geräteelementen. Die Netzwerkparameter werden im Netzwerkobjekt gespeichert. Die Parameter, die eine Netzwerkschnittstelle eines bestimmten Geräteelements betreffen, das mit einem Netzwerk verbunden ist, werden in einem Netzwerknotenobjekt in diesem Geräteelement gespeichert. Die Kommunikation wird oft mit „Kanälen“, „Ports“ und „Schnittstellen“ geregelt.

Subnetze werden als interne Elemente mit der Rollenklassifizierung „Subnetz“ in der Instanzhierarchie in der AML-Datei exportiert.

Ein Subnetz hat folgende verbundene Elemente in der AML-Struktur:

- Internes Element mit der Rollenklassifizierung „Knoten“
Legt die Schnittstelle auf einem Geräteelement fest.
- <InternalLink>
Legt die verbundenen Partner des Subnetzes fest. Der <InternalLink> Variablenname ist einzigartig und wird immer unter dem internen Element des Projekts in der AML-Datei hinzugefügt.

```

.....
<SupportedRoleClass RefRoleClassPath=
  "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
<InternalLink Name="Link To Port_1"
  RefPartnerSideA="1e3e4c5b-04c1-4d2c-9aee-cad53cc92dba:CommunicationPortInterface"
  RefPartnerSideB="d45aa36a-a7f2-4862-a266-d6727b9cfd75:CommunicationPortInterface" />
<InternalLink Name="Link To Subnet_1"
  RefPartnerSideA="beb4eb8e-1a45-45ce-a703-1acfac73e5f3:LogicalEndPoint_Node"
  RefPartnerSideB="1062a384-d3ca-4183-9ac2-0934a5ab7286:LogicalEndPoint_Subnet" />
<InternalLink Name="Link To Subnet_2"
  RefPartnerSideA="a3e85aed-580a-45c8-943e-da7de8280b7c:LogicalEndPoint_Node"
  RefPartnerSideB="1062a384-d3ca-4183-9ac2-0934a5ab7286:LogicalEndPoint_Subnet" />
</InternalElement>
</InstanceHierarchy>
</CAEXFile>

```

- <ExternalInterface>
Stellt in internen Elementen von Knoten und Subnetzen fest, dass Knoten und Subnetze verbunden sind. Wenn die Knoten oder Subnetze nicht verbunden sind, existieren die <ExternalInterface> Elemente für Knoten und Subnetz nicht.

```

...
<InternalElement ID="1062a384-d3ca-4183-9ac2-0934a5ab7286" Name="PN/IE_1">
  <Attribute Name="Type" AttributeDataType="xs:string">
    <Value>Ethernet</Value>
  </Attribute>
  <ExternalInterface ID="55ecf2cb-e72a-4974-8ed0-1d4e1ade3509"
    Name="LogicalEndPoint_Subnet"
    RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Subnet" />
</InternalElement>
...

```

Verwendung

In TIA Portal V16 unterstützt der CAx-Import Ethernet und Profinet und der Export erfolgt für die Attribute "Subnetztyp" und die "Knotentyp" immer über Ethernet.

Das rechnergestützte Importieren/Exportieren unterstützt folgende Arten von Subnetzen:

- Ethernet
- PROFIBUS
- MPI
- ASi

Attribute eines „Subnetz“-Elements

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	
Type	Obligatorisch	Ethernet oder PROFIBUS oder MPI oder ASi

Attribute von Elementen "CommunicationInterface"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte für CAX-Import- und -Exportdateien:

Attribut	Verarbeitung	Kommentar
Name	Obligatorisch	Keine Relevanz bei "festen" Geräteelementen.
Label	Obligatorisch	Beschriftung fehlt möglicherweise, wenn "BuiltIn" = TRUE und "PositionNumber" für das zugehörige Objekt "DeviceItem" angegeben sind.
TypeIdentifier	Obligatorisch	Für integrierte Geräteelemente muss dieses Attribut mit der Typkennung seines steckbaren übergeordneten Elements exportiert werden. Während des Imports ist dieses Attribut nicht relevant. Bei nicht integrierten Geräteelementen hat dieses Attribut keine Relevanz.
FirmwareVersion	Obligatorisch	
TypeName	Nur Export	Keine Relevanz bei "integrierten" Geräteelementen.
DeviceItemType	Nur Export	Nur für CPU und Kopfmodul Dieses Attribut ist beim Import optional, doch jedes Geräteelement außer Basiseinheiten mit DeviceItemType als Zubehör wird stillschweigend ignoriert.
PositionNumber	Obligatorisch	Keine Relevanz beim Import von "integrierten" Geräteelementen.
BuiltIn	Obligatorisch für Export Optional für Import	Keine Relevanz für den Import von "nicht integrierten" Geräteelementen. Für den Import standardmäßig "False".
Comment	Optional	Nicht zutreffend bei "integrierten" Geräteelementen.

Attribute von Elementen "CommunicationPort"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte für CAx-Import- und -Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	Keine Relevanz bei "integrierten" Geräteelementen.
Label	Obligatorisch	Das Attribut 'Label' besitzt keine Relevanz für nicht integrierte Port-Geräteelemente und wird daher nicht exportiert. Ab TIA Portal V19 enthält der Wert von 'Label' beim Exportieren der Port-Konfiguration kein Leerzeichen (Beispiel: P1R). Beim Importieren des Werts für ein Port-Label wird beim Vergleich des Label-Werts nicht nach Groß- und Kleinschreibung unterschieden (z. B.: P1R, P1, p1). Bei Ports, die für eine Ringtopologie freigegeben und mit Suffix "R" gekennzeichnet sind, ist der Import "gleichwertig" mit Ports zu behandeln, deren Label-Werte kein Suffix enthalten. Beispiel: Die folgenden Port-Label werden als gleich behandelt: • P1 ist gleich P1 • P1 ist gleich P1R • P1R ist gleich P1 • P1R ist gleich P1R
Typidentifizier	Obligatorisch	Das Attribut ist für den Export und Import von nicht integrierten Geräteelementen obligatorisch. Beispiel: Geräteelemente mit Integriert = False Für integrierte Geräteelemente muss dieses Attribut mit der Typkennung seines steckbaren übergeordneten Elements exportiert werden. Während des Imports ist dieses Attribut nicht relevant. Für nicht integrierte Geräteelemente ist dieses Attribut nicht relevant.
FirmwareVersion	Obligatorisch	Dies ist ein obligatorisches Attribut für den Export und Import dann und nur dann, wenn das Geräteelement nicht integriert ist und die Firmwareversion unterstützt. Das Attribut ist nicht anwendbar, wenn das Geräteelement integriert ist.
TypeName	Obligatorisch für Export Optional beim Import	Keine Relevanz bei "integrierten" Geräteelementen.
PositionNummer	Obligatorisch	Das Attribut ist für den Export und Import aller Typen von Port-Geräteelementen obligatorisch. Beim Import ist das Attribut obligatorisch für nicht integrierte Port-Geräteelemente, besitzt aber keine Relevanz für integrierte Port-Geräteelemente
BuiltIn	Obligatorisch für Export Optional für Import	Keine Relevanz für den Import von "nicht integrierten" Geräteelementen. Für den Import standardmäßig "False".
Comment	Optional	Nicht zutreffend bei "integrierten" Geräteelementen.

Attribute eines „Knoten“-Elements

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Nur Export	MPI, PROFIBUS, PROFINET
Type	Nur Exportieren	Ethernet oder PROFIBUS oder MPI oder ASI
NetworkAddress	Obligatorisch	
SubnetMask	Optional	PROFINET Beim Import wird der Standardwert beibehalten, wenn kein Wert festgelegt ist.
RouterAddress	Optional	PROFINET Beim Import wird der Standardwert beibehalten, wenn kein Wert festgelegt ist.
DhcpClientId	Optional	PROFINET Beim Import wird der Standardwert beibehalten, wenn kein Wert festgelegt ist.
IpProtocolSelection	Optional	PROFINET Beim Import wird der Standardwert beibehalten, wenn kein Wert festgelegt ist. Werte: Project, Dhcp, UserProgram, OtherPath, Address Tailoring

Bei dem Profinet-Knoten weicht die Verfügbarkeit von Attributen basierend auf bestimmten Vorbedingungen ab. Beispiel:

- Wenn in der UI des TIA Portal als IP-Protokoll "Set IP address in the project" ausgewählt ist, dann sind "NetworkAddress" und "SubnetMask" verfügbar.
- Das Attribut "RouterAddress" muss für den Export/Import verfügbar sein, wenn als IP-Protokoll "Set IP address in the project" und in der UI "Use router" ausgewählt sind.
- Das Attribut "DhcpClientId" ist nur dann für den Export/Import verfügbar, wenn das Attribut 'IpProtocolSelection' 'Dhcp' ist.

Sonderfall CAX-Import, bei dem das Attribut 'Type' erforderlich ist - Profibus-Subnetzverbindung wird durch Ändern des Schnittstellentyps in der UI von 'MPI' in 'PROFIBUS' hergestellt.

- Im TIA Portal ist für bestimmte Geräteelemente der Schnittstellentyp 'Mpi' voreingestellt. Dieser Typ muss jedoch in 'Profibus' geändert werden, um eine Profibus-Subnetzverbindung herzustellen. In diesem Fall muss als Typ der Wert 'Profibus' in der AML-Datei angegeben werden, um die Typkonvertierung beim Import zu zeigen.

Hinweis

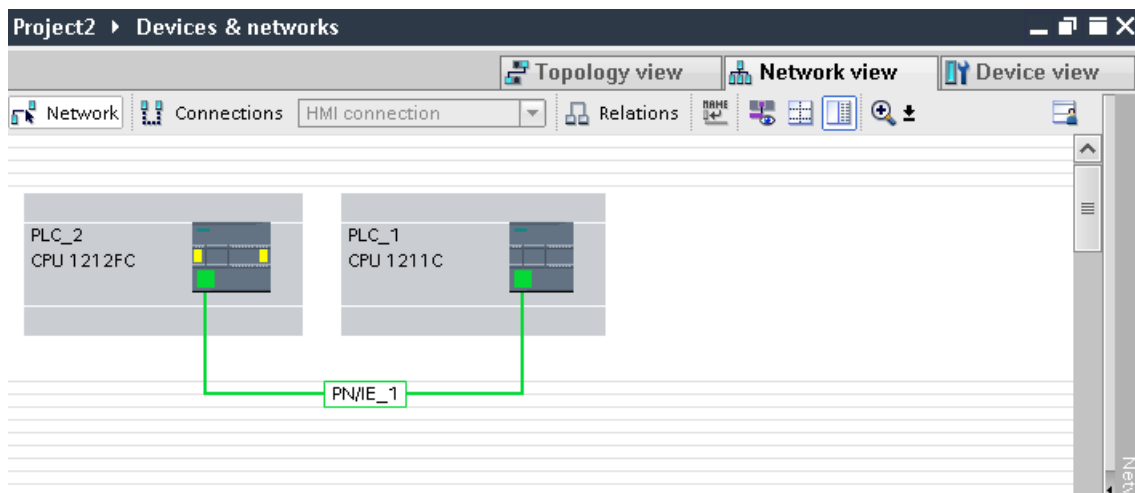
- Der CAX-Import unterstützt Ethernet und Profinet für die Attribute Subnetztyp und Knotentyp.
- Der CAX-Import einer AML-Datei mit Stringwerten ohne Unterscheidung von Groß-/Kleinschreibung für die Attribute (Beispiel: Profinet, PROFINET, pROFINET, ProFINET usw.) wird unterstützt.
- Der CAX-Export erfolgt immer über Ethernet.

Attribute eines „Kanal“-Elements

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Type	Obligatorisch	Digital oder analog
IoType	Obligatorisch	Eingang oder Ausgang
Number	Obligatorisch	
Length	Nur Exportieren	

Beispiel: Exportiertes Subnetz



AML-Struktur

Die folgenden Abbildungen zeigen die Struktur der exportierten AML-Datei.

```
...
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
  <InternalElement ID="e9d2bedb-f8c1-4148-acda-c3c68836c7dd" Name="Project2">
    ...
    <InternalElement ID="1062a384-d3ca-4183-9ac2-0934a5ab7286" Name="PN/IE_1">
      <Attribute Name="Type" AttributeDataType="xs:string">
        <Value>Ethernet</Value>
      </Attribute>
      <ExternalInterface ID="55ecf2cb-e72a-4974-8ed0-1d4e1ade3509"
        Name="LogicalEndPoint_Subnet"
        RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
      <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Subnet" />
    </InternalElement>
    <InternalElement ID="b011dbb1-efa4-46c0-a26f-f9bd047cda4f" Name="S7-1200_station_1">
      ...
      <InternalElement ID="d006e41b-05ff-44ab-baab-fca15f99e86c" Name="PROFINET interface_1">
        ...
        <InternalElement ID="beb4eb8e-1a45-45ce-a703-1acfac73e5f3" Name="E1">
          ...
          <ExternalInterface ID="a365b498-20cc-4e0b-99ca-5c5257632b96"
            Name="LogicalEndPoint_Node" RefBaseClassPath=
              "CommunicationInterfaceClassLib/LogicalEndPoint" />
          <SupportedRoleClass RefRoleClassPath=
            "AutomationProjectConfigurationRoleClassLib/Node" />
        </InternalElement>
        <InternalElement ID="d45aa36a-a7f2-4862-a266-d6727b9cfd75" Name="Port_1">
          ...
          <ExternalInterface ID="32c6ba4a-b01f-4678-b721-ea284779e96c"
            Name="CommunicationPortInterface"
            RefBaseClassPath=
              "AutomationProjectConfigurationInterfaceClassLib/CommunicationPortInterface" />
          <SupportedRoleClass RefRoleClassPath=
            "AutomationProjectConfigurationRoleClassLib/CommunicationPort" />
        </InternalElement>
        <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
        </InternalElement>
        <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
        </InternalElement>
        <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
        </InternalElement>
        <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/Device" />
        </InternalElement>
      </InternalElement>
    </InternalElement>
  </InternalElement>
</InstanceHierarchy>
...
```

```

...
<InternalElement ID="7cf0ea2b-b66f-4ad4-8a03-5a8691cbe04d" Name="PLC_2">
...
<InternalElement ID="b287020d-667b-483d-a8e0-c5466ac2f5c3" Name="PROFINET interface_1">
  <Attribute Name="Label" AttributeDataType="xs:string">
    <Value>X1</Value>
  </Attribute>
  <InternalElement ID="a3e85aed-580a-45c8-943e-da7de8280b7c" Name="E1">
    <Attribute Name="Type" AttributeDataType="xs:string">
      <Value>Ethernet</Value>
    </Attribute>
    <Attribute Name="NetworkAddress" AttributeDataType="xs:string">
      <Value>192.168.0.2</Value>
    </Attribute>
    <Attribute Name="SubnetMask" AttributeDataType="xs:string">
      <Value>255.255.255.0</Value>
    </Attribute>
    <Attribute Name="DeviceNumber" AttributeDataType="xs:string">
      <Value>0</Value>
    </Attribute>
    <Attribute Name="IpProtocolSelection" AttributeDataType="xs:string">
      <Value>Project</Value>
    </Attribute>
    <ExternalInterface ID="6ae8eb93-09d3-4f8c-b529-a12148c71bf4"
      Name="LogicalEndPoint_Node"
      RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
    <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
  </InternalElement>
  <InternalElement ID="1e3e4c5b-04c1-4d2c-9aee-cad53cc92dba" Name="Port_1">
    ...
    <ExternalInterface ID="1f5b2a3d-fcd1-460a-b846-30dadc8726d1"
      Name="CommunicationPortInterface"
      RefBaseClassPath="
        AutomationProjectConfigurationInterfaceClassLib/CommunicationPortInterface" />
    <SupportedRoleClass RefRoleClassPath="
      AutomationProjectConfigurationRoleClassLib/CommunicationPort" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath="
    AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="
  AutomationProjectConfigurationRoleClassLib/DeviceItem" />
</InternalElement>
...

```

Erweiterte Rolle für Profinet/Ethernet-Knoten

Im Rahmen der jüngsten Empfehlung von AR APC 1.2 müssen alle Profinet/Ethernet-Knoten gegen einen Knoten mit der zusätzlichen Rolle 'AutomationProjectConfigurationRoleClassLib/NodeEthernet' zusammen mit der vorhandenen Rolle 'AutomationProjectConfigurationRoleClassLib/Node' ausgetauscht werden, sofern erweiterte Attribute unterstützt werden.

Der Profinet/Ethernet-Knoten muss mit der zusätzlichen Rolle exportiert werden, sofern er erweiterte Attribute wie SubnetMask, RouterAddress, DhcpClientId, IpProtocolSelection unterstützt. Darüber hinaus muss es möglich sein, den Knoten mit oder ohne erweiterte Rolle zu importieren.

Der nachfolgende Auszug aus der XML-Datei zeigt die AML-Datei mit dem Knoten "Ethernet" mit erweiterter Rolle.

```
<InternalElement ID="9c4393a9-44d5-49b4-9314-02eb0f94b6c0" Name="IE1">
<Attribute Name="SubnetMask" AttributeDataType="xs:string">
<Value>255.255.255.0</Value>
</Attribute>
<Attribute Name="RouterAddress" AttributeDataType="xs:string">
<Value>192.168.0.1</Value>
</Attribute>
<Attribute Name="IpProtocolSelection" AttributeDataType="xs:string">
<Value>Project</Value>
</Attribute>
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Ethernet</Value>
</Attribute>
<Attribute Name="NetworkAddress" AttributeDataType="xs:string">
<Value>192.168.0.1</Value>
</Attribute>
<ExternalInterface ID="eea59d3c-3bc4-4d0d-9815-46b4b347369d" Name="LogicalEndPoint_Node"
RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationEthernetRoleClassLib/
NodeEthernet" />
</InternalElement>
```

Siehe auch

- Struktur der CAx-Daten zum Import/Export (Seite 1482)
- Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen (Seite 131)

6.5.33 Export/Import von IO-Systemen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist offline.

AML-Struktur

IO-Systeme werden in der AML-Struktur als `<InternalElement>` dargestellt.

IO-Systeme eines Masters oder IO-Controllers werden unter dem Element `<CommunicationInterface>` des Geräteelements einer Schnittstelle hinzugefügt.

```

...
<InternalElement ID="[Communication Interface UniqueID]"
  Name="[Communication Interface Name]">
...
  <!--Node-->
  <InternalElement ID="[Node UniqueID]" Name="[Node Name]">
    ...
    <ExternalInterface ID="[External Interface UniqueID]"
      Name="LogicalEndPoint_Node"
      RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
    <SupportedRoleClass
      RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
  </InternalElement>
  <!--IoSystem-->
  <InternalElement ID="[IoSystem UniqueID]" Name="[IoSystem Name]">
    <Attribute Name="Number" AttributeDataType="xs:integer">
      <Value>[IoSystem Number]</Value>
    </Attribute>
    <ExternalInterface ID="[External Interface UniqueID]"
      Name="LogicalEndPoint_Interface"
      RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
    <SupportedRoleClass
      RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/IoSystem" />
  </InternalElement>
  <SupportedRoleClass
    RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
</InternalElement>

```

Als Slave oder IO-Device verbundene IO-Systeme werden als Elemente `<ExternalInterface>` unter dem Element `<CommunicationInterface>` des Geräteelements einer Schnittstelle hinzugefügt.

```

<InternalElement ID="[Communication Interface UniqueID]"
  Name="[Communication Interface Name]">
...
  <ExternalInterface ID="[External Interface UniqueID]"
    Name="LogicalEndPoint_Interface"
    RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
  <!--Node-->
  <InternalElement ID="[Node UniqueID]" Name="[Node Name]">
    <ExternalInterface ID="[External Interface UniqueID]"
      Name="LogicalEndPoint_Node"
      RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
    <SupportedRoleClass
      RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
  </InternalElement>
  <SupportedRoleClass
    RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
</InternalElement>

```

Die verbundenen Partner des IO-Systems werden als Elemente `<InternalLink>` dargestellt. Die Variablen `<InternalLink>` werden unter dem gemeinsamen übergeordneten Element eines IO-Systems und des verbundenen Slave-Geräteelements hinzugefügt, z. B. unter `Project`, `DeviceFolder`, `DeviceItem`.

Der Variablenname <InternalLink> ist im gemeinsamen übergeordneten Element eindeutig.

Attribute eines Elements "IO-System"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	Der Name des IO-Systems. Wird ein leerer String importiert, wird das IO-System mit dem Standardnamen erstellt.
Number	Optional	Erfolgt keine Angabe für den Import, wird der Standardwert verwendet.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.34 Exportieren/Importieren mehrsprachiger Kommentare

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist offline.

Verwendung

Export- und Importkommentare und mehrsprachige Kommentare zum CAX-Datenaustausch der folgenden Hardwareobjekte:

- Geräte (Gerät)
- Module (Geräteelemente)
- Variablen (Variable)

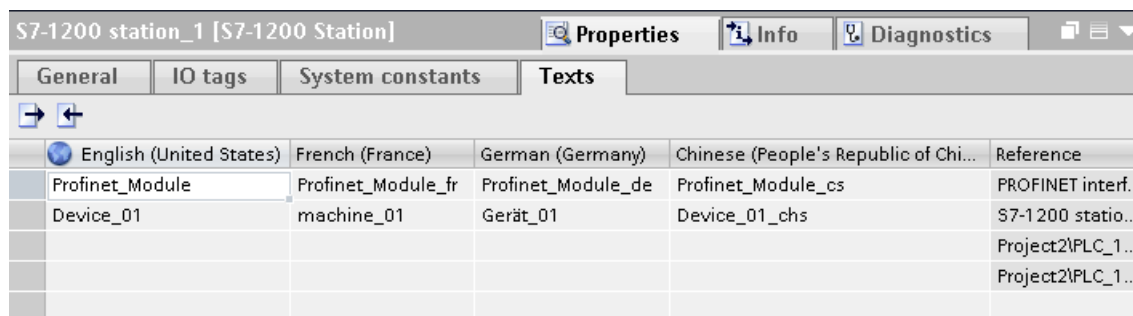
Das Importieren/Exportieren mehrsprachiger Kommentare umfasst alle Sprachen des TIA Portals.

Einschränkungen

- Export
 - Nur, wenn ein Kommentar existiert, wird ein Attribut „Comment“ in die AML-Datei exportiert.
- Import
 - Das Attribut "Comment" ist optional.
 - Für virtuelle Geräteelemente kann kein Kommentar importiert werden.

Beispiel: Exportierte Konfiguration mit mehrsprachigen Kommentaren

Das folgende Bild zeigt die Konfiguration eines SIMATIC S7 1500 (Gerät) mit PLC_1 (Geräteelemente). Bei beiden Objekten werden Kommentare in Englisch, Französisch, Deutsch und Chinesisch festgelegt.



English (United States)	French (France)	German (Germany)	Chinese (People's Republic of Chi...)	Reference
Profinet_Module	Profinet_Module_fr	Profinet_Module_de	Profinet_Module_cs	PROFINET interf...
Device_01	machine_01	Gerät_01	Device_01_chs	S7-1200 statio...
				Project2\PLC_1 ...
				Project2\PLC_1 ...

AML-Struktur

Nach dem Exportieren dieser Konfigurationen, werden die mehrsprachigen Kommentare als verschachtelte Attribute des Geräts, Geräteelements oder der Variablen generiert.

- Das übergeordnete Attribut "Comment" muss den in der Standardsprache verwendeten Wert besitzen.
- Das Nachfolgerattribut existiert für jeden fremdsprachigen Kommentar.

```

...
<Attribute Name="Comment" AttributeDataType="xs:string">
  <Value>English</Value>
  <Attribute Name="aml-lang=en-US" AttributeDataType="xs:string">
    <Value>English</Value>
  </Attribute>
  <Attribute Name="aml-lang=de-DE" AttributeDataType="xs:string">
    <Value>Deutsch</Value>
  </Attribute>
  <Attribute Name="aml-lang=zh-HK" AttributeDataType="xs:string">
    <Value>Chinese</Value>
  </Attribute>
  ...
  ...
</Attribute>
...

```

Siehe auch

- Struktur der CAx-Daten zum Import/Export (Seite 1482)
- Verbindung zum TIA Portal aufbauen (Seite 84)
- Projekt öffnen (Seite 131)

6.5.35 Export/Import von ET200 SP/ET200 AL-Geräten mit Erweiterungsbaugruppenträgerverbindungen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt wird geöffnet
Siehe Öffnen eines Projekts (Seite 131)

Anwendung

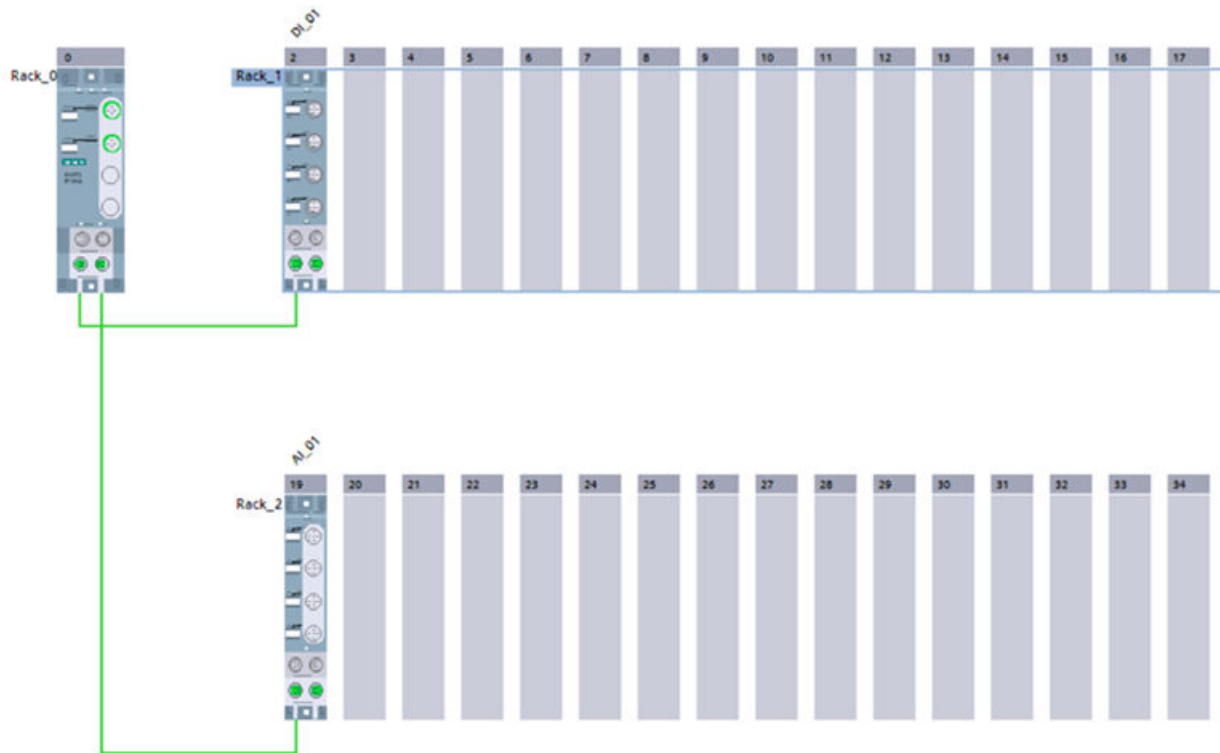
Es muss möglich sein, ein Gerät mit mehreren Baugruppenträgern mit Erweiterungsbaugruppenträgerverbindungen in eine AML-Datei zu exportieren und wieder zu importieren, um das gleiche Gerät mit Erweiterungsbaugruppenträgerverbindungen im TIA Portal-Projekt wieder zu erstellen.

In TIA Portal werden Erweiterungsbaugruppenträgerverbindungen zwischen mehreren Baugruppenträgern direkt unter DeviceItem-Objekten (ET-Con_1, ET-Con_2 und ET-Connection Receiver) modelliert. Entsprechend der AR APC-Empfehlung werden diese Verbindungen jedoch als Port-zu-Port-Verbindungen unter einem Kommunikationsport modelliert.

Gemäß dieser Empfehlung wird deshalb ein Dummy CommunicationInterface mit Dummy CommunicationPort-Objekten unter den jeweiligen DeviceItem-Objekten hinzugefügt.

Gerätekonfigurationen mit mehreren Erweiterungsbaugruppenträgerverbindungen

Die folgende Konfiguration zeigt ein ET-200AL-Gerät mit mehreren Baugruppenträgern mit Erweiterungsbaugruppenträgerverbindungen in TIA Portal.



AML-Export

Die AML-Datei, die beim Export für obige Konfiguration erzeugt werden soll, ist im Folgenden abgebildet.

6.5 Hardware-Daten importieren/exportieren

```

<?xml version="1.0" encoding="utf-8"?><CAEXFile FileName="ET200AL_with_ExtensionRacks.aml"
SchemaVersion="2.15" xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
...
<InternalElement ID="2b90fale-df69-437d-8a77-eea455cdfaba" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ET-Con</Value>
</Attribute>
...
<InternalElement ID="32758129-15db-43c7-9deb-53613852b208" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X30</Value></Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="c326d6b8-cc1a-444b-8a06-5c63a226a456"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
...
<InternalElement ID="ab4e0e97-8e57-4218-b55c-0e4fe89a9587" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ET-Con</Value>
</Attribute>
...
<InternalElement ID="6bb61ad0-13ac-455b-9a90-6cbaddeaabd4" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X31</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="90c95545-2e43-4701-8fd0-9e87ef0bd412"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<InternalElement ID="54119ad3-34b6-4e63-bc18-bee23312900a" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">

```

```

<Value>ET-Con</Value>
</Attribute>
...
<InternalElement ID="ad638a2a-538e-4840-9d99-7712522431ba" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X30</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="6a517987-ae22-4446-b361-91aea2c768be"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
...
<InternalLink Name="Link To Port_1"
RefPartnerSideA="32758129-15db-43c7-9deb-53613852b208:CommunicationPortInterface"
RefPartnerSideB="ad638a2a-538e-4840-9d99-7712522431ba:CommunicationPortInterface" />
<InternalLink Name="Link To Port_2"
RefPartnerSideA="6bb61ad0-13ac-455b-9a90-6cbaddeaabd4:CommunicationPortInterface"
RefPartnerSideB="cac8af19-67bc-41aa-a3f0-7149af8f67d4:CommunicationPortInterface" />

```

Hinweis

Die Erweiterungsbaugruppenträgerschnittstelle wird nur dann exportiert, wenn Erweiterungsbaugruppenträgerverbindungen vorhanden sind. Außerdem richtet sich die Anzahl der Dummy-Ports unterhalb der ET-Con-Dummy-Erweiterungsschnittstelle nach der Anzahl der Ports, die auf Modulebene zur Baugruppenträgerverbindung gehören.

Erweiterungsbaugruppenträgerverbindung

Die XML-Darstellung von Erweiterungsbaugruppenträgerverbindungen zwischen mehreren Baugruppenträgern erfolgt im unten dargestellten Format.

- **ExternalInterface** Das interne Element -<ExternalInterface> muss unter dem internen Element <CommunicationPort> hinzugefügt werden, das Bestandteil der Verbindung ist.

```

<InternalElement ID="[IM Module Unique ID]" Name="[IM Module Name]">
<InternalElement ID="[Dummy Interface Unique ID]" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ET-Con</Value>
</Attribute>
...
<InternalElement ID="[Dummy Port Unique ID]" Name="[IM Module Sender/Receiver Name]">
...
<ExternalInterface ID="[External Interface Unique ID]" Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="[Dummy Port Unique ID]" Name="[IM Module Sender/Receiver Name]">
...
<ExternalInterface ID="[External Interface Unique ID]" Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>

```

InternalLink- Erweiterungsbaugruppenträgerverbindungen werden mithilfe von <InternalLink>-Variablen dargestellt. <InternalLink>-Variablen müssen unter dem gemeinsamen übergeordneten Element der verschiedenen Baugruppenträger (z. B. Gerät) hinzugefügt werden. Der Name des internen Links muss im gemeinsamen übergeordneten Element eindeutig sein.

```

<InternalLink Name="Link To [Internal link Name]" RefPartnerSideA="[Communication Port
UniqueID]:[Communication Port External Interface Name]" RefPartnerSideB="[Communication
Port UniqueID]:[Communication Port External Interface Name]" />

```


AML-Import

Es wird möglich sein, Details zu Erweiterungsbaugruppenträgerverbindungen aus einer AML-Datei zu importieren, die beim oben erwähnten Export erzeugt wurde. Es wird jedoch auch möglich sein, AML-Dateien zu importieren, die in früheren Versionen von TIA Portal erzeugt wurden.

Hinweis

1. Das veränderte Verhalten der Exporthierarchie gilt erst ab Version V16. Ältere Versionen von TIA Portal behalten das gleiche Verhalten wie zuvor.
 2. Die Hierarchie in der AML-Datei beeinflusst/beeinträchtigt die interne Hierarchie in TIA Portal nach dem Import nicht.
 3. Diese Änderung des Verhaltens der Hierarchie gilt für ET200 AL und ET200 SP mit Erweiterungsbaugruppenträgerverbindungen.
 4. Auch die AML-Datei mit Stringwerten ohne Unterscheidung von Groß-/Kleinschreibung für das Attribut Type (z. B. ET-Con, et-con, ET-con usw.,,) wird fehlerfrei importiert.
-

Erweiterungsbaugruppenträger mit Reihenverbindungen

Die folgende Konfiguration zeigt einen Erweiterungsbaugruppenträger mit in Reihe nebeneinander angeordneten Baugruppen. In der nachstehend dargestellten Konfiguration sind die Baugruppen DI_01, DI_02, DI_03, AQ_01 und AQ_02 jeweils verbunden mit ihren benachbarten Baugruppen in TIA Portal dargestellt. In TIA Portal wird die Verbindung zwischen dem BusAdapter-Sender und den DI_01-Baugruppen als Erweiterungsbaugruppenträgerverbindung modelliert, die Verbindungen anderer Baugruppen werden nicht modelliert.



Die AML-Datei, die beim Export für die obige Konfiguration erzeugt werden soll, ist im Folgenden abgebildet.

6.5 Hardware-Daten importieren/exportieren

```

<?xml version="1.0" encoding="utf-8"?><CAEXFile
FileName="ET200SP_with_ET200AL_ExtensionRack_And_SideBySideConnection.aml"
SchemaVersion="2.15" xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
...
<InternalElement ID="aa0f752d-63fb-4f8d-b840-72d39f2ae508" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ET-Con</Value>
</Attribute> ...
<InternalElement ID="695f110e-2c2b-435f-b3c1-cdc77d83d559" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute> <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="87b9a1a3-dab8-4c62-8627-fbc8ab84e8f7"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement><SupportedRoleClass
RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
</InternalElement>
<InternalElement ID="124371c4-52ca-4ecc-a476-84e7bc1439b8" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ET-Con</Value>
</Attribute>
...
<InternalElement ID="0e354480-219c-4178-9f21-21fb369d446e" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X30</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="ab03743a-59a5-490e-b6c6-ce5359e345fd"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
...
</InternalElement>
<InternalElement ID="a0ff4884-cb81-40ed-85c8-823650687a8f" Name="Port_2">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X31</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>3</Value>

```

```

</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="33caac7c-2944-4b1d-9fb8-8a451e4c1c25"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<InternalElement ID="d1aa8afe-e20a-4b82-996a-9fca151b2c51" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ET-Con</Value>
</Attribute>
...
<InternalElement ID="e713beda-1d1c-48d7-8674-8640080c87f7" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X30</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="046e2a6a-dca9-4a63-8f4f-53cabac1a65c"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="81999cc0-694f-4f7b-ba12-72ee9010cb11" Name="Port_2">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X31</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>3</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="e611d310-9ff7-45bb-b379-36e9cbea5080"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>

```

6.5 Hardware-Daten importieren/exportieren

...

```
<InternalElement ID="d21e9e3d-f5ea-4992-87b4-1598ecd2e159" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>ET-Con</Value>
</Attribute>
```

...

```
<InternalElement ID="74cff4f7-3023-4235-9255-b36bc9b3e3e6" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X30</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="5322297a-a9c6-474d-82d0-992234730926"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
```

...

```
<InternalElement ID="c671263b-4c6e-4914-8a29-cec88bc917bd" Name="RackExtension">
<Attribute Name="Type" AttributeDataType="xs:string"><
Value>ET-Con</Value>
</Attribute>
```

...

```
<InternalElement ID="b220aab7-ed61-481e-a9e5-5588b510c46c" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X30</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<ExternalInterface ID="ef830254-217a-42b3-b2f1-2363acf79b69"
Name="CommunicationPortInterface"
RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/
CommunicationPortInterface" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<InternalLink Name="Link To Port_1" RefPartnerSideA="695f110e-2c2b-435f-b3c1-
cdc77d83d559:CommunicationPortInterface"
RefPartnerSideB="0e354480-219c-4178-9f21-21fb369d446e:CommunicationPortInterface" />
```

```

<InternalLink Name="Link To Port_2" RefPartnerSideA="a0ff4884-
cb81-40ed-85c8-823650687a8f:CommunicationPortInterface"
RefPartnerSideB="e713beda-1d1c-48d7-8674-8640080c87f7:CommunicationPortInterface" />
<InternalLink Name="Link To Port_3" RefPartnerSideA="81999cc0-694f-4f7b-
ba12-72ee9010cb11:CommunicationPortInterface" RefPartnerSideB="74cff4f7-3023-4235-9255-
b36bc9b3e3e6:CommunicationPortInterface" />
<InternalLink Name="Link To Port_4"
RefPartnerSideA="0201304e-39a9-4c9a-8316-2dd86fd95d0c:CommunicationPortInterface"
RefPartnerSideB="b220aab7-ed61-481e-a9e5-5588b510c46c:CommunicationPortInterface" />
...

```

Hinweis

Bei Reihenverbindungen muss die Erweiterungsbaugruppenschchnittstelle nur exportiert werden, wenn Baugruppen nebeneinander gesteckt sind. Außerdem richtet sich die Anzahl der Dummy-Ports unterhalb der ET-Con-Dummy-Erweiterungsschnittstelle nach der Anzahl der Ports, die auf Modulebene zur Baugruppenträgerverbindung gehören. Im Beispiel oben ist Baugruppe DI_02 zwischen den Baugruppen DI_01 und DI_03 gesteckt, in der exportierten AML-Datei hat die entsprechende ET-Con-Dummy-Schnittstelle somit zwei Dummy-Ports. Dagegen hat Baugruppe AI_01 keine direkt benachbarten Baugruppen. Für die Baugruppe AI_01 existieren daher keine ET-Con-Dummy-Schnittstellendetails. Wird die AML-Datei wieder in ein leeres TIA Portal-Projekt importiert, wird das gleiche Gerät mit mehreren Baugruppenträgern mit Erweiterungsbaugruppenträgerverbindungen erstellt.

- Für die Baugruppen ET200 AL und ET200SP unterstützt TIA Portal Standardverbindungen zwischen den primären und den Erweiterungsbaugruppenträgern, wenn die jeweiligen Baugruppen gesteckt werden. In diesem Fall sind die Verbindungsinformationen aus der AML-Datei beim Import nicht relevant.
- Beschreibungen von Reihenanordnungen der Verbindungen in der AML-Datei haben beim Import keine Relevanz, da Reihenverbindungen automatisch eingerichtet werden, wenn zwei benachbarte Baugruppen in den gleichen Baugruppenträger importiert werden.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.36 Exportieren/Importieren von PLC-Variablen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Die PLC ist offline.

Verwendung

Exportierte und importierte Symbole und Variablen werden einem Geräteelement zugeordnet. Rechnergestütztes Importieren/Exportieren betrifft hardwarenahe Symbole und Variablen. Die Symbole und Variablen werden nur mit dem Geräteelement des Steuersollwerts exportiert, z. B. der CPU und nicht mit anderen Geräteelementen, auf die sie sich beziehen, z. B. einem E/A-Modul. Wie Geräte, werden die Variablen oft in Variablentabellen und in einer hierarchischen Ordnerstruktur unterteilt.

AML-Strukturelemente

PLC-Variablen, Variablentabellen und Variablen-Nutzerordner können durch die rechnergestützte Import-/Exportfunktion exportiert und importiert werden. Die Variablen-Objekte werden in den folgenden AML-Strukturelementen abgebildet:

- `<InternalElement>`
Variablentabellen und Variablen-Nutzerordner werden als interne Elemente der zugehörigen PLC mit der jeweiligen Rollenklassifizierung abgebildet.
- `<ExternalInterface>`
Stellt eine PLC-Variable dar, die zu dem internen Element der zugehörigen Variablentabelle oder dem Variablen-Nutzerordner gehört.

Ein Zuordnungskanal mit einer PLC-Variable wird über das Element `<internal link>` als Kommunikationspartner exportiert. Die folgende XML-Struktur zeigt ein Beispiel:

```
...
<InternalLink Name="Link To Tag_1"
  RefPartnerSideA="b33451f6-d88f-4900-8dbe-41f1be1e3535:Channel_DI_0"
  RefPartnerSideB="b2b937ee-d5db-4826-9340-027b1da22828:Tag_1" />
...
```

PLC-Variablen-Nutzerordner

Die Objekte „TagUserFolder“ benötigen nur das Attribut „Name“ in rechnergestützten Import- und Exportdateien.

Attribute einer PLC-Variablentabelle

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte rechnergestützter Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch, wird ignoriert, wenn „AssignToDefault“ = TRUE:	
AssignToDefault	Nur Importieren	Wird verwendet, um die Standardvariablentabelle während des Imports zu identifizieren. Wenn "AssignToDefault" = TRUE, werden alle Variablen im Rahmen der Standardvariablentabelle des TIA Portals erstellt.

Attribute einer PLC-Variablen

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	
DataType	Obligatorisch	
LogicalAddress	Optional	Wird im internationalen Mnemonik-Format importiert und exportiert
IoType	Optional	Eingang oder Ausgang
Comment	Optional	

Hinweis

TIA Portal V16 unterstützt den Export und Import des Attributs IoType in AML-Dateien mit AR APC V1.1.0.

Beispiel: AML-Struktur

Die folgende Abbildung zeigt die Struktur der folgenden exportierten Variablenobjekte:

- leere Standardvariablentabelle
- Variablen-Nutzerordner „Gruppe_1“
- enthaltene Variablentabelle „Variablentabelle_1“
- vier Variablen

```
<InternalElement ID="a310b193-ba04-49d7-a8e3-004619c7d9d2" Name="Default tag table">
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/TagTable" />
</InternalElement>
<InternalElement ID="0feff703-9c70-4ca9-b3b3-8de8229696dd" Name="Group_1">
  <InternalElement ID="f92g9ce4-c015-459f-9f59-8f94bca3b186" Name="Tag table_1">
    <ExternalInterface ID="fc0c8c5a-fd5b-443b-b430-6435b6aa22ff" Name="Tag_1" RefBaseClassPath="AutomationProjectConfigurationRoleClassLib/TagTable" />
    ...
    </ExternalInterface>
    <ExternalInterface ID="450d6a1d-81b8-49ae-a104-c0072933d669" Name="Tag_2" RefBaseClassPath="AutomationProjectConfigurationRoleClassLib/TagTable" />
    ...
    </ExternalInterface>
    <ExternalInterface ID="3de17a36-b5c5-4fc7-9fc3-47e4a8f95087" Name="Tag_3" RefBaseClassPath="AutomationProjectConfigurationRoleClassLib/TagTable" />
      <Attribute Name="DataType" AttributeDataType="xs:string">
        <Value>Word</Value>
      </Attribute>
      <Attribute Name="IoType" AttributeDataType="xs:string">
        <Value>Input</Value>
      </Attribute>
      <Attribute Name="LogicalAddress" AttributeDataType="xs:string">
        <Value>IWO</Value>
      </Attribute>
    </ExternalInterface>
    <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/TagTable" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/TagUserFolder" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/TagUserFolder" />
</InternalElement>
...
```

Siehe auch

Struktur der CAx-Daten zum Import/Export (Seite 1482)

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.37 Export/Import von RH/PLC**Voraussetzung**

- TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Der PLC ist offline.

Anwendung

Sie können mit dem TIA Portal AML-Dateien nach AR APC V1.1 mit R/H PLC in gleicher IP-Adresskonfiguration importieren.

Attribut

TIA Portal unterstützt nur ein Attribut in der AML-Datei für R/H PLC, falls in der Benutzeroberfläche des TIA Portals verfügbar:

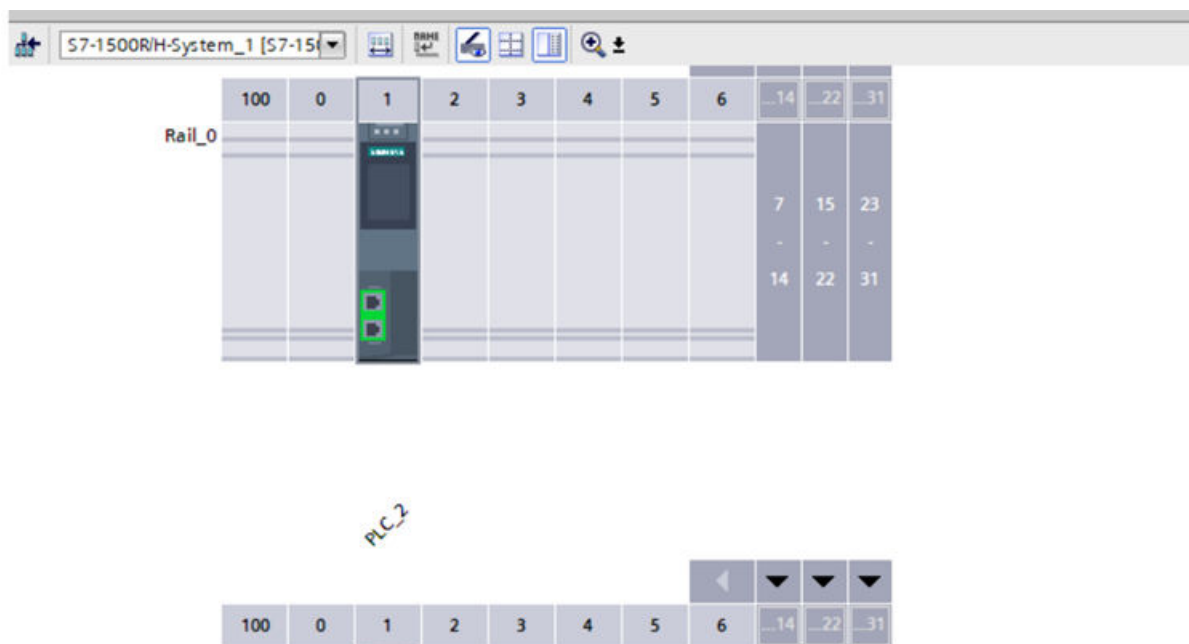
Attributname des Geräteelements	Handhabung	Kommentar
HNetworkAddress	Obligatorisch	Exportiert/Importiert, wenn der R/H PLC des Geräteelements dieses Attribut in TIA Portal unterstützt wird und nicht leer ist, andernfalls wird er übersprungen. Für den Import wird "System IP-Adresse für geschaltete Verbindung freigeben" eingestellt und die HNetworkAddress bewertet.

Einschränkung

- Exportieren
 - Nur bei Auswahl von "System IP-Adresse für geschaltete Verbindung freigeben" in TIA Portal.

Beispiel: Exportierte Konfiguration

Die folgende Konfiguration zeigt ein Geräteelement, bei dem die HNetworkAddress konfiguriert ist.



AML-Struktur der Exportdatei

Die folgende Abbildung zeigt die Struktur der exportierten AML-Datei für R/H PLC.

```
<InternalElement ID="e1879cf8-8222-4ce3-b171-60c56aed7f18" Name="PROFINET interface_1">
  <Attribute Name="Label" AttributeDataType="xs:string">
    <Value>X1</Value>
  </Attribute>
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>32768</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>true</Value>
  </Attribute>
  <InternalElement ID="a32c67c7-47dc-4362-a9c4-b41b93b58eff" Name="E1">
    <Attribute Name="Type" AttributeDataType="xs:string">
      <Value>Ethernet</Value>
    </Attribute>
    <Attribute Name="NetworkAddress" AttributeDataType="xs:string">
      <Value>192.168.0.1</Value>
    </Attribute>
    <Attribute Name="SubnetMask" AttributeDataType="xs:string">
      <Value>255.255.255.0</Value>
    </Attribute>
    <Attribute Name="IpProtocolSelection" AttributeDataType="xs:string">
      <Value>Project</Value>
    </Attribute>
    <Attribute Name="HNetworkAddress" AttributeDataType="xs:string">
      <Value>192.168.0.3</Value>
    </Attribute>
  </InternalElement>
  <ExternalInterface ID="802676fa-212f-4bf5-b112-de94993a0340" Name="LogicalEndPoint_Node"
    RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
</InternalElement>
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.38 Export/Import von AML mit benutzerdefinierten Variablen und Geräteelementen

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Der PLC ist offline.

Anwendung

Sie können mit dem TIA Portal AML-Dateien nach AR APC V1.1 mit einem Subattribut des Typs 'Customized' für Variable und DeviceItem exportieren und importieren.

Attribut

Die folgende Tabelle zeigt das zugehörige Attribut, das für eine Variable beim CAX-Import und -Export von Dateien verfügbar ist:

Attributname	Handhabung	Kommentar
Customized	Optional beim Export/Import	Subattribut von 'DataType' einer Variablen. Für 'Customized' sind nur die Werte TRUE und FALSE erlaubt. Während des Imports ist das Attribut 'Customized' nicht relevant. Während des Exports hat das Subattribut 'Customized' bei allen Datentypen, die nicht IEC 61131 entsprechen, den Wert "True". Bei Datentypen nach IEC 61131 wird das Subattribut 'Customized' nicht exportiert.

Die folgende Tabelle zeigt das zugehörige Attribut, das für ein Geräteelement beim CAX-Import und -Export von Dateien verfügbar ist:

Attributname	Handhabung	Kommentar
Customized	Optional beim Import	Es ist ein Nachfolgeattribut des übergeordneten Attributs 'DeviceItem' für ein Geräteelement (DeviceItem). Bei Customized sind nur die Werte TRUE und FALSE erlaubt. Während des Imports und Exports ist das Attribut Customized nicht relevant.

Exportierte AML-Datei

Die folgende AML-Datei wird während des Exports einer AML-Datei mit Geräteelementen mit Attribut 'Customized' generiert.

```
<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="Project26.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
<AdditionalInformation>
<WriterHeader>
...
</WriterHeader>
</AdditionalInformation>
<AdditionalInformation AutomationMLVersion="2.0" />
<AdditionalInformation DocumentVersions="Recommendations">
<Document DocumentIdentifier="AR APC" Version="1.1.0" />
</AdditionalInformation>
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
<InternalElement ID="03ecf798-3e07-4976-b281-f8b98eb3a590" Name="Project26">
...
<InternalElement ID="c218aea9-93b8-4719-be6f-ccfa9517e2c6" Name="S71500/ET200MP station_1">
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>System:Device.S71500</Value>
</Attribute>
<InternalElement ID="clae306f-183f-47b8-91e6-cb331c559278" Name="Rail_0">
...
<InternalElement ID="8d7d5ee1-603a-4897-b47e-8954d4c21a31" Name="PLC_1">
...
<Attribute Name="DeviceItemType" AttributeDataType="xs:string">
<Value>CPU</Value>
</Attribute>
...
<InternalElement ID="a2a7f0bc-068c-4904-84b1-73ed87e28de1" Name="Tag table_1">
...
<Attribute Name="DataType" AttributeDataType="xs:string">
<Value>Conn_Any</Value>
<Attribute Name="Customized" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
</Attribute>
</InternalElement>
...
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
Device" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
AutomationProject" />
</InternalElement>
</InstanceHierarchy>
</CAEXFile>
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.39 Import/Export fehlersicherer PLCs**Voraussetzung**

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Der PLC ist offline.

Anwendung

Sie können mit dem TIA Portal eine AML-Datei nach AR APC V1.1 mit fehlersicherer PLC mit fehlersicheren Attributen exportieren und importieren.

Attribute

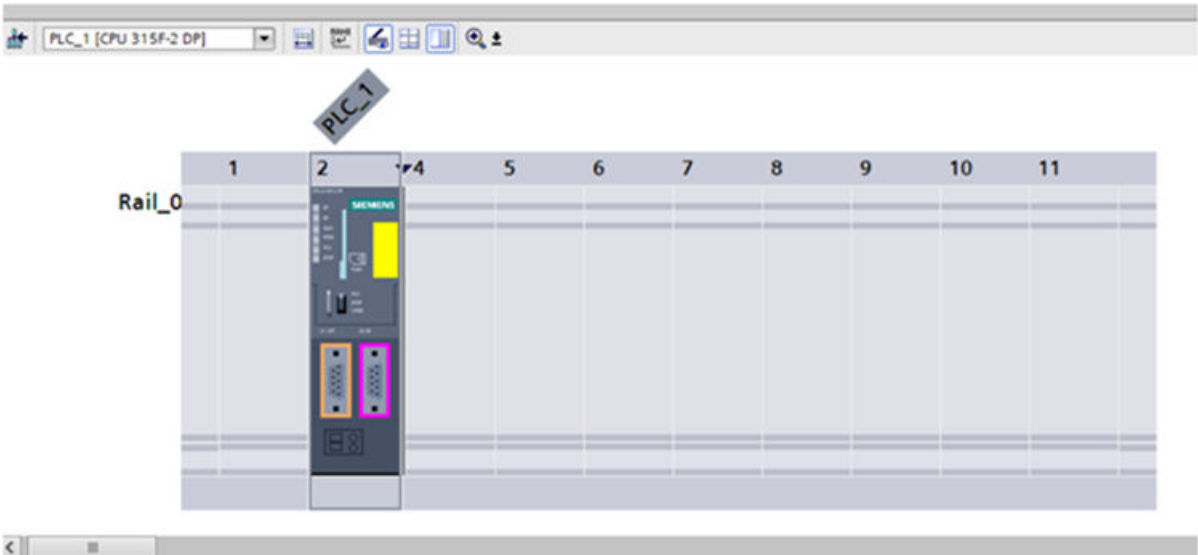
Die folgende Tabelle zeigt die Liste der fehlersicheren Attribute von AML-Dateien für den CAX-Import und CAX-Export:

Attribute in Openness	Handhabung	Kommentar	AR APC-Name in AML
Failsafe_FSourceAddress	Obligatorisch	Export/Import nur, wenn das Geräteelement ein fehlersicherer PLC ist und dieses Attribut in TIA Portal unterstützt und nicht leer ist, andernfalls wird er übersprungen.	Failsafe_FSourceAddress
Failsafe_LowerBoundForFDestinationAddresses	Obligatorisch	Export/Import nur, wenn das Geräteelement ein fehlersicherer PLC ist und dieses Attribut in TIA Portal unterstützt und nicht leer ist, andernfalls wird er übersprungen.	Failsafe_LowerBoundForFDestinationAddresses
Failsafe_UpperBoundForFDestinationAddresses	Obligatorisch	Export/Import nur, wenn das Geräteelement ein fehlersicherer PLC ist und dieses Attribut in TIA Portal unterstützt und nicht leer ist, andernfalls wird er übersprungen.	Failsafe_UpperBoundForFDestinationAddresses

Attribute in Openness	Handhabung	Kommentar	AR APC-Name in AML
Failsafe_CentralFSourceAddress	Optional	Export/Import nur, wenn das Geräteelement ein fehlersicherer PLC ist und dieses Attribut in TIA Portal unterstützt und nicht leer ist, andernfalls wird er übersprungen.	Failsafe_FSourceAddress
Failsafe_FDestinationAddress	Optional	Export/Import nur, wenn das Geräteelement ein fehlersicherer PLC ist und dieses Attribut in TIA Portal unterstützt und nicht leer ist, andernfalls wird er übersprungen.	Failsafe_FDestinationAddress

Beispiel: Exportierte Konfiguration

Die folgende Konfiguration zeigt ein Geräteelement, bei dem die Attribute konfiguriert sind.



AML-Struktur der Exportdatei

Die folgenden Abbildungen zeigen die Struktur der exportierten AML-Datei.

```
<InternalElement ID="9c944d2c-e0ae-4f39-b35a-a63faaf35be7" Name="PLC_1">
  <Attribute Name="TypeName" AttributeDataType="xs:string">
    <Value>CPU 1511TF-1 PN</Value>
  </Attribute>
  <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
    <Value>CPU</Value>
  </Attribute>
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>1</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>false</Value>
  </Attribute>
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>OrderNumber:6ES7 511-1UK01-0AB0</Value>
  </Attribute>
  <Attribute Name="InstallationDate" AttributeDataType="xs:dateTime">
    <Value>2019-02-28T08:12:12.987Z</Value>
  </Attribute>
  <Attribute Name="Failsafe_FSourceAddress" AttributeDataType="xs:string">
    <Value>1</Value>
  </Attribute>
  <Attribute Name="Failsafe_LowerBoundForFDestinationAddresses"
    AttributeDataType="xs:string">
    <Value>1</Value>
  </Attribute>
  <Attribute Name="Failsafe_UpperBoundForFDestinationAddresses"
    AttributeDataType="xs:string">
    <Value>99</Value>
  </Attribute>
  <Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
    <Value>V2.8</Value>
  </Attribute>
</InternalElement>
<InternalElement ID="4f718e93-b541-4983-8f13-1f5b21c3e70c" Name="Default tag table">
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
    TagTable"/>
</InternalElement>
<InternalElement ID="20e19f5b-8ace-4e0c-af0b-c710ae4817da" Name="CPU display_1">
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>3</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>true</Value>
  </Attribute>
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
    DeviceItem" />
</InternalElement>
<InternalElement ID="5a24516f-17d6-4b2a-a4ac-efc1b577875d" Name="Card reader/writer_1">
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>4</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>true</Value>
  </Attribute>
```

```

<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement> <InternalElement ID="0f746d71-035e-4e64-b0d7-51d0449cfd88" Name="OPC
UA_1">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>254</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value> </Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<InternalElement ID="a0633104-a2ac-4680-bb99-81df50f5ec40" Name="PROFINET interface_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>X1</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>32768</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<InternalElement ID="e3497176-dbbba-4fec-9d3a-772ae13987c4" Name="E1">
<Attribute Name="Type" AttributeDataType="xs:string">
<Value>Ethernet</Value> </Attribute>
<Attribute Name="NetworkAddress" AttributeDataType="xs:string">
<Value>192.168.0.1</Value>
</Attribute>
<Attribute Name="SubnetMask" AttributeDataType="xs:string">
<Value>255.255.255.0</Value>
</Attribute>
<Attribute Name="IpProtocolSelection" AttributeDataType="xs:string">
<Value>Project</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
</InternalElement>
<InternalElement ID="3208384f-d5ba-4ccb-b8da-f08ec38ec681" Name="Port_1">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P1R</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>32769</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<InternalElement ID="4a47c05e-9656-4e02-9b51-23b065b6fe47" Name="Port_2">
<Attribute Name="Label" AttributeDataType="xs:string">
<Value>P2R</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>32770</Value>
</Attribute>

```

```

<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationPort" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
CommunicationInterface" />
</InternalElement>
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
</InternalElement>
<InternalElement ID="e3fdb611-4b68-4682-b154-ae43c74a24d3" Name="F-DI 16x24V DC_1">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>F-DI 16x24V DC</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>>false</Value> </Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 526-1BH00-0AB0</Value>
</Attribute>
<Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
<Value>V1.0</Value>
</Attribute>
<InternalElement ID="77c4fea0-baba-44e6-80f2-72b7b830a88a" Name="F-DI 16x24V DC_1">
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>1</Value>
</Attribute> <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>true</Value>
</Attribute>
<Attribute Name="Failsafe_FSourceAddress" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="Failsafe_FDestinationAddress" AttributeDataType="xs:string">
<Value>655</Value>
</Attribute>
</InternalElement>
</InternalElement>

```

Erweiterte Rolle für Geräteelement

AR APC 1.2 empfiehlt die folgenden Änderungen für den CAX-Austausch im TIA Portal ab V17 aufwärts: jedes Geräteelement muss mit einer zusätzlichen Rolle ausgetauscht werden.

'AutomationProjectConfigurationProfiSafeRoleClassLib/DeviceItemProfiSafe' mit der vorhandenen Rolle 'AutomationProjectConfigurationRoleClassLib/DeviceItem', sofern von AR APC empfohlene fehlersichere Attribute unterstützt werden.

Das Geräteelement muss mit der zusätzlichen Rolle exportiert werden, sofern es erweiterte Attribute wie die folgenden unterstützt: Failsafe_FSourceAddress, Failsafe_LowerBoundForFDestinationAddresses, Failsafe_UpperBoundForFDestinationAddresses, Failsafe_CentralFSourceAddress,

Failsafe_FDestinationAddress. Darüber hinaus muss es möglich sein, das Geräteelement mit oder ohne erweiterte Rolle zu importieren.

AML-Datei mit Geräteelement

Der nachfolgende Auszug aus der XML-Datei zeigt die AML-Datei mit dem "Geräteelement", das Module fehlersicherer Attribute mit erweiterter Rolle unterstützt.

```
<InternalElement ID="9d6c270a-2a48-426a-9dae-8cf88c5a591a" Name="PLC_1">
<Attribute Name="TypeName" AttributeDataType="xs:string">
<Value>CPU 414F-3 PN/DP</Value>
</Attribute>
<Attribute Name="DeviceItemType" AttributeDataType="xs:string">
<Value>CPU</Value>
</Attribute>
<Attribute Name="PositionNumber" AttributeDataType="xs:int">
<Value>2</Value>
</Attribute>
<Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
<Value>>false</Value>
</Attribute>
<Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
<Value>OrderNumber:6ES7 414-3FM06-0AB0</Value>
</Attribute>
<Attribute Name="Failsafe_FSourceAddress" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="Failsafe_LowerBoundForFDestinationAddresses"
AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="Failsafe_UpperBoundForFDestinationAddresses"
AttributeDataType="xs:string">
<Value>99</Value>
</Attribute>
<Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
<Value>V6.0</Value>
</Attribute>
...
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
DeviceItem" />
<SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationProfiSafeRoleClassLib/
DeviceItemProfiSafe" />
</InternalElement>
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.40 Import/Export fehlersicherer E/A

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Öffnen eines Projekts (Seite 131)
- Der PLC ist offline.

Anwendung

Sie können mit dem TIA Portal eine AML-Datei nach AR APC V1.1 mit fehlersicheren E/A mit fehlersicheren Attributen exportieren und importieren.

Attribute

Die folgende Tabelle zeigt die zugehörigen Attribute, die an fehlersicheren E/A-Modulen/-Submodulen für CAX-Importdateien und CAX-Exportdateien verfügbar sind:

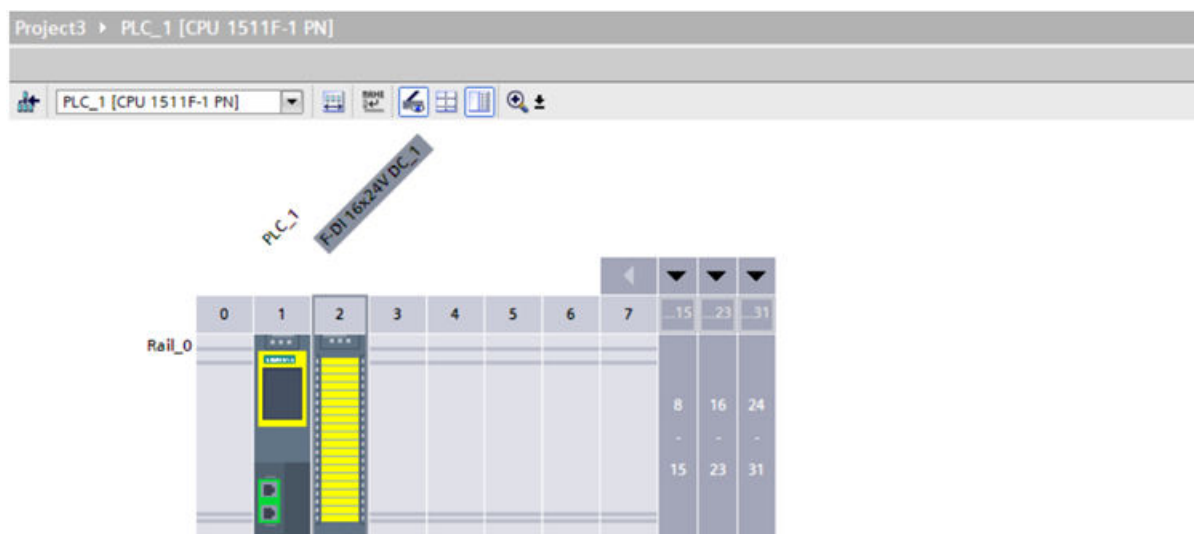
Attributname des Geräteelements	Handhabung	Kommentar
FSourceAddress	Obligatorisch	Export/Import nur, wenn das Geräteelement fehlersichere E/A ist und in TIA Portal unterstützt wird.
FDestinationAddresses	Obligatorisch	Export/Import nur, wenn das Geräteelement fehlersichere E/A ist und in TIA Portal unterstützt wird.

Hinweis

- Für die meisten fehlersicheren E/A ist FSourceAddress schreibgeschützt – also nicht beschreibbar und eine Reflexion des Attributs FCentralFSourceAddress des fehlersicheren PLC. Es wird der gleiche Wert exportiert und beim Import ignoriert.
 - Wenn FSourceAddress und FDestinationAddress schreibgeschützt sind, wird der Wert exportiert. Jedoch muss während des Imports eine Warnmeldung im Info-Register des TIA Portals angezeigt werden.
 - FSourceAddress und FDestinationAddress müssen auf E/A-Modul- und E/A-Submodulebene unterstützt werden.
-

Beispiel: Exportierte Konfiguration

Die folgende Konfiguration zeigt ein Geräteelement, bei dem die Attribute konfiguriert sind.



AML-Struktur der Exportdatei

Die folgende Abbildung zeigt die Struktur der exportierten AML-Datei.

```
<Attribute Name="FSourceAddress" AttributeDataType="xs:string">
<Value>1</Value>
</Attribute>
<Attribute Name="FDestinationAddress" AttributeDataType="xs:string">
<Value>65534</Value>
</Attribute>
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.41 Herstellerspezifisches Attribut exportieren/importieren

Voraussetzung

- Die Anwendung TIA Portal Openness ist mit dem TIA Portal verbunden.
Siehe Verbindung zum TIA Portal herstellen (Seite 84)
- Ein Projekt ist geöffnet.
Siehe Projekt öffnen (Seite 131)

Anwendung

Im TIA Portal können Sie eine AML-Datei für Geräte und Geräteelemente mit dem Attribut "Manufacturer" exportieren und importieren, so dass Sie herstellerspezifische Informationen in Round-Trip-Szenarien austauschen können.

Das Attribut "Manufacturer" wird von Openness nicht unterstützt. Es wird beim Export/Import vom "Transformation-Plugin" aufgenommen. AML-Dateien mit dem Empfehlungsdokument zu AR Drive ab Version 1.0 werden unterstützt.

Hinweis

AML-Dateien müssen von TIA Portal V16 nach AR APC V1.1 generiert/exportiert werden.

Exportieren

Sie können das Attribut "Manufacturer" in eine AML-Datei exportieren, wenn es im AmiModel verfügbar ist, das von Nutzern des "Transformation-Plugin" ausgefüllt wird.

Importieren

Sie können das Attribut "Manufacturer" ins TIA Portal importieren, in dem das herstellerspezifische Attribut im AmiModel verfügbar ist, das von Nutzern des "Transformation-Plugin" aufgenommen wird.

Derzeit wird dieses Attribut nur für Geräte und Geräteelemente vom Typ "Antrieb" unterstützt, da nur diese das offizielle, für CAX geschriebene "Transformation-Plugin" haben.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 84)

Projekt öffnen (Seite 131)

6.5.42 AML-Attribute im Vergleich zu TIA Portal Openness-Attributen

Auf Attribute und Export-/Importattribute zugreifen

Über TIA Portal Openness können Sie auf Hardware-Attribute von Hardwareobjekten zugreifen. Einzelne Namen, die Sie für den Zugriff auf diese Attribute verwenden, z. B. von einem Geräteelement, unterscheiden sich von den Attributnamen in der Export-/Import-AML-Datei.

Liste der Attribute

Die folgende Tabelle bietet einen Überblick über beide Arten von Attributen:

Tabelle 6-6 Attributnamen von Geräten und GSD/GSDML-Geräten

AML-Datei	TIA Portal Openness
Name	Name
Typidentifizier	Typidentifizier
Comment	Comment

Tabelle 6-7 Attributnamen von Geräteelementen

AML-Datei	TIA Portal Openness
Name	Name
Typidentifizier	Zugeordnet zum Substring von <code><TypeIdentifier></code> (d.h. Wert vor ersten Operator <code>" "</code>), wobei der Bestandteil mit der Firmwareversion ignoriert wird. Die Zuordnung des Substrings ist nur zutreffend, wenn die Typkennung mit dem Präfix <code><OrderNumber:></code> beginnt und einen Bestandteil mit der Firmwareversion hat, ansonsten erfolgt die Zuordnung zur Vervollständigung von <code><TypeIdentifier></code> .
FirmwareVersion	<code><FirmwareVersion></code> ist dem Substring von <code><TypeIdentifier></code> zugeordnet (d. h. Wert nach dem ersten Operator <code>" "</code>). Die Zuordnung des Substrings ist nur zutreffend, wenn <code><TypeIdentifier></code> mit dem Präfix <code><OrderNumber:></code> beginnt und einen Bestandteil mit der Firmwareversion hat.
TypeName	TypeName
DeviceItemType (für CPU und Kopfmodul)	Classification
PositionNumber	PositionNumber
BuiltIn	IsBuiltIn
PlantDesignation IEC	PlantDesignation
LocationIdentifier IEC	LocationIdentifier
Comment	Comment

Tabelle 6-8 Attributnamen von GSD/GSDML-Geräteelementen

AML-Datei	TIA Portal Openness
Name	Name
TypenIdentifier	TypenIdentifier
TypeName	TypeName
DeviceItem Type (für Kopfmodul)	Classification
PositionNumber	PositionNumber
BuiltIn	IsBuiltIn
Comment	Comment
Label	Label

Tabelle 6-9 Attributnamen von Variablen

AML-Datei	TIA Portal Openness
Name	Name
DataType	DataTypeName
LogicalAddress	LogicalAddress
Comment	Comment

Tabelle 6-10 Attributnamen von Variablentabellen

AML-Datei	TIA Portal Openness
Name	Name
AssignToDefault	IsDefault

Tabelle 6-11 Attributnamen von Adressen

AML-Datei	TIA Portal Openness
StartAddress	StartAddress
Length	Length
IoType	IoType

Tabelle 6-12 Attributnamen von Ports

AML-Datei	TIA Portal Openness
Name	Name
TypenIdentifier	TypenIdentifier
FirmwareVersion	FirmwareVersion
TypeName	TypeName
PositionNumber	PositionNumber
BuiltIn	IsBuiltIn

AML-Datei	TIA Portal Openness
Comment	Comment
Label	Label

Tabelle 6-13 Attributnamen von Geräten mit IO-Schnittstelle

AML-Datei	TIA Portal Openness
Name	Name
Typeldentifier	Typeldentifier
FirmwareVersion	FirmwareVersion
TypeName	TypeName
DeviceItemTyp (für CPU und Kopfmodul)	Classification
PositionNumber	PositionNumber
BuiltIn	IsBuiltIn
Label	Label
Comment	Comment

Tabelle 6-14 Attributnamen von Kanälen

AML-Datei	TIA Portal Openness
Type	Type
IoType	IoType
Number	Keinem Attribut in TIA Portal Openness zugeordnet.
Length	ChannelWidth

Wesentliche Änderungen

7.1 Größere Änderungen für Langzeitstabilität in TIA Portal Openness V19

Änderungen

Wenn Sie die Hinweise zur versionsübergreifenden Programmierung beachtet haben und Ihre Openness-Anwendung nicht auf V19 aufbauen, läuft Ihre Anwendung ohne jede Einschränkung auf jedem Rechner, auch wenn nur ein TIA Portal V19 installiert ist.

Wenn Sie Ihre Openness-Anwendung auf V19 aufbauen, ist es notwendig, Ihre Anwendung mit der Siemens.Engineering.dll von V19 neu zu übersetzen. In manchen Fällen kann es erforderlich sein, den Code Ihrer Anwendung anzupassen.

CAx/AML-Datenaustausch

Änderungen im Verhalten der Befehlszeilenfunktion

Ab TIA Portal V19 ist die Befehlszeilenfunktion für den CAx-Export und -Import für den Benutzer nicht mehr verfügbar.

Änderung im Verhalten des PROFINET/Ethernet-Port-Handlings

Ab TIA Portal V19 wird der Labelwert für PROFINET/Ethernet-Schnittstellenports ohne Leerzeichen exportiert, unabhängig von seinem Wert in TIA Portal. Beispiel: Ist das Label = P1 R, hätte die AML-Datei das Label = P1R

Änderung im Verhalten des IO Link-Port-Handlings

Ab TIA Portal V18 Update 2 wird der IO-Link-Port mit einem geänderten Labelwert 'C/Q<n>' ausgetauscht, wobei n die Portnummer ist. Beispiel: C/Q1, C/Q2 usw.
Verwenden Sie für den AML-Austausch der IO-Link-Konfiguration in TIA Portal V18 Update 2 (über S7-PCT) die Version "S7-PCT 3.5 SP3 Update 3" oder höher.

WebServerUserManagement

Änderungen im Verhalten beim Entfernen/Deaktivieren von vorhandenen Openness-Diensten

Bei TIA Portal V19 ist die zentralisierte UMAC-Funktion aktiv, aber WebServerUserManagement ist für alle PLCs mit einer Firmware-Version ab V3.1 deaktiviert.

Bis TIA Portal V18 ist jedoch WebServerUserManagement betriebsbereit, während die zentralisierte UMAC-Funktion für PLCs mit einer Firmware-Version ab V3.0 inaktiv ist.

UMAC auf PLC

Änderungen im Verhalten eines Online-Legitimationsaufrufs über UMAC auf PLC

Wenn die Openness-Anwendung mit einer TIA Portal Openness API ≤ 18 läuft, ist die Online-Legitimation über Openness über die Download- und Upload-Konfiguration noch möglich. Dies gilt nur, wenn die verwendete PLC durch eine alte ProtectionLevel-Legitimation geschützt ist.

Bei TIA Portal Openness API $\geq V19$ werden alle Legitimationsaufrufe, insbesondere das neue UMAC auf PLC, dem Event Handler des Online-Legitimationsereignisses der ConnectionConfiguration-Klasse bereitgestellt. Sollte der Typ OnlineAuthenticationConfiguration dort nicht vom Benutzercode behandelt werden, erfolgt ein zweiter Aufruf der Rückrufmethode der jeweiligen Funktion (z. B. Download oder Upload). Dort können jedoch nur die alten Schutzmechanismen (bis V18) verarbeitet werden.

Durchsetzung strenger Passwort-Richtlinien

Änderungen im Verhalten bei strengen Passwort-Richtlinien für den Know-how-Schutz von Bausteinen

Bei TIA Portal V19 werden strenge Passwort-Richtlinien bei der Einstellung eines Passworts angewandt. Dies gilt auch für Aufrufe von TIA Portal Openness, insbesondere für die Einstellung des Know-how-Schutzes. Andernfalls wird eine Ausnahme ausgelöst, wenn das Passwort die Richtlinie nicht erfüllt.

WinCC Unified-Bildeditor

Änderungen im Verhalten von MultilingualText-Elementen im Rahmen des WinCC Unified-Bildeditors

Alle MultilingualText-Elemente werden im Rahmen des WinCC Unified-Bildeditors seit TIA Portal V18 von unformatiert in formatiert geändert.

Das bedeutet, dass alle Texte von nun an formatiert gesetzt werden müssen. Klartext wird durch Auslösen einer Ausnahme abgelehnt. Beispiel:

```
Language language = project.LanguageSettings.Languages.Find(new CultureInfo("en-US"));
MultilingualText multilingualToolTipText1 = ((HmiButton)screenItem1).ToolTipText;
MultilingualTextItem multilingualTextItem1 = multilingualToolTipText1.Items.Find(language);
multilingualTextItem1.Text = "<body><p>Modified button text from Openness</p></body>";
```

Unterstützung von Bausteinen/UDTs

Änderungen im Schema, um die Verwendung von NamedValueTypes in Bausteinen zu unterstützen

Im TIA Portal V19 wurde die Unterstützung für genannten Werttypen eingeführt (nur für Software Units in S7-1500 PLCs).

In Openness wird ein neuer Gültigkeitsbereich "NamedValueConstant" im SimaticML-Schema ab V19 eingeführt, um die Verwendung eines genannten Werttyps in Programmbausteinen oder PLC-Datentypen zu unterstützen.

Die genannten Werttypen, die in PLC-Programmierartefakten (Programmbausteinen, PLC-

Datentypen) verwendet werden, sind im neuen Gültigkeitsbereich "NamedValueConstant" beim Importieren/Exportieren über SimaticML sichtbar.

```
<Access Scope="NamedValueConstant" UId="27">
<Constant Name="_.siemens.simatic.Named_value_type_1#UNDEFs" UId="28"/>
</Access>
<Token Text=";" UId="31"/>
<NewLine Num="2" UId="32"/>
```

7.2 Größere Änderungen für Langzeitstabilität in TIA Portal Openness V17

Änderungen

Wenn Sie die Hinweise zur versionsübergreifenden Programmierung beachtet haben und Ihre Openness-Anwendung nicht auf V17 aufbauen, laufen Ihre Anwendungen ohne jede Einschränkung auf jedem Rechner, auch wenn nur ein TIA Portal V17 installiert ist.

Wenn Sie Ihre Openness-Anwendung auf V17 aufbauen, ist es notwendig, Ihre Anwendung mit der SiemensEngineering.dll von V17 neu zu übersetzen. In manchen Fällen kann es erforderlich sein, den Code Ihrer Anwendung anzupassen.

Änderung bei der Implementierung von GetAttribute

Bis TIA Portal Openness V16 war die Sichtbarkeit der Implementierung von GetAttribute/SetAttribute/GetAttributeInfos davon abhängig, ob für den Typ "Unterstützt dynamisches Attribut" deklariert wurde. Für den Zugriff auf Methoden wie GetAttribute/SetAttribute/GetAttributeInfos sollte der Typ dynamische Attribute unterstützen. Wenn der Typ keine dynamischen Attribute unterstützt, müssen Sie explizit an IEngineeringObject übertragen, um die Anzeige der Methoden GetAttribute/SetAttribute/GetAttributeInfos zu sehen.

Das folgende Beispiel zeigt, wie der Code bis TIA Portal Openness V16 ausgeführt wird:

```
var projectAttributeInfos = ((IEngineeringObject) project).GetAttributeInfos();
```

In TIA Portal Openness V17 werden alle Methoden von IEngineeringObject implizit in allen Typen implementiert, die IEngineeringObject implementieren, der explizite Aufruf ist somit nicht mehr erforderlich und alle Methoden werden angezeigt. Die Änderung steht für alle früheren Engineering-dlls wie V15, V15.1 & V16 zur Verfügung, die in V17 unterstützt werden.

Für die folgenden Methoden ist eine Typumwandlung nicht erforderlich:

- **object GetAttribute(string name)** – ruft ein Attribut mit dem angegebenen Namen ab.
- **IList<EngineeringAttributeInfo> GetAttributeInfos()** – gibt eine Sammlung von EngineeringAttributeInfo-Objekten zurück, die die verschiedenen Attribute zu diesem Objekt beschreiben.

- **IList<object> GetAttributes(IEnumerable<string> names)** – ruft eine Liste von Attributen für die angegebenen Namen ab.
- **void SetAttribute(string name, object value)** – legt ein Attribut mit dem angegebenen Namen auf den angegebenen Wert fest.
- **void SetAttributes(IEnumerable<KeyValuePair<string, object>> attributes)** – legt die Attribute mit den angegebenen Namen auf die angegebenen Werte wie in den Attributen angegeben fest.

Mit dieser Änderung verbunden ist ein geringfügig verändertes Verhalten beim erneuten Kompilieren des Codes, der unter der Methode Extension verwendet wird. Wurde beispielsweise die Methode Extension verwendet und der Code wird in die V17 Engineering-dll kompiliert, wird ein Aufruf aus Extension nicht ausgeführt. Sie müssten in diesem Fall den Code so ändern, dass er mit dem Verhalten kompatibel bleibt. Es würde dann auch kein Compiler-Fehler auftreten.

Der folgende Code zeigt den Fall, in dem die Methode GetAttribute nicht ausgeführt wird:

```
static class CustomerExtension
{
    public static object GetAttribute(this Project project, string name)
    {
        // Customer Logic
        return ((IEngineeringObject)project);
    }
}
```

Änderungen in AssignInterface API

Vor TIA Portal Openness V17 kann AssignInterface API nicht in Transaktionen in TIA Portal Openness verwendet werden. Ab TIA Portal Openness V17 ändert sich dies und AssignInterface API kann jetzt in Transaktionen verwendet werden.

Einstellung der Mastersystem-Nummer

Ab TIA Portal Openness V17 kann die Nummer des Mastersystems beim Importieren einer API eingestellt werden.

Entfernung der Methode ExternalSourceCreate

Vor TIA Portal Openness V17 wurde die Erstellung aus einer externen Quelle mit "Nicht unterstützte Ausnahme" in TIA Portal Openness unterstützt. Ab TIA Portal Openness V17 wird das Erstellen aus externen Quellen entfernt.

Entfernung von Attribut SyncRole

Vor TIA Portal Openness V17 war das Attribut "SyncRole" für das Gerät IPC627D schreibgeschützt. Ab TIA Portal Openness V17 wird das Attribut "SyncRole" von Geräten entfernt, wo das Attribut keinen Zweck erfüllt.

Geändertes Verhalten von Safety Compile

Vor TIA Portal Openness V17 wird Failsafe mit einem festgelegten Passwort und ohne angemeldeten Benutzer nicht unterstützt und gibt einen Übersetzungsfehler zurück. Ab TIA Portal Openness V17 ändert sich dies und statt eines Übersetzungsfehlers wird eine Ausnahme zurückgegeben.

Entfernung der Ladekonfiguration

Ab TIA Portal Openness V17 sind STEP 7-spezifische Ladekonfigurationen nicht mehr in TIA Portal Openness verfügbar, wenn STEP 7 nicht installiert ist.

Einschränkung beim Attribut "SecondaryType" beim Import von OB MC-Transformation

Bei TIA Portal Openness V17 muss beim Import des OB MC-Transformation das Attribut "SecondaryType" den exakten Zeichenkettenwert "Transformation" haben (das ist auch der Wert, der exportiert wird).

ProDiag-Informationen exportieren

Ab TIA Portal Openness V17 können Sie ProDiag-Meldungen mit einem ProDiag-FB über die öffentliche Openness API in das CSV-Format exportieren.

Die Ausgabe kann in Microsoft Excel angesehen werden.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Identification	Priority	Alarm text	Information text	Additional text 1	Additional text 2	Additional text 3	Additional text 4	Additional text 5	Additional text 6	Additional text 7	Additional text 8	Additional text 9
2	1	0	1	SubCategory1	SubCategory2	Message	PLC_E	Default_SupervisionFB	2	1	%B 0 - globalTag	tag comment	@%1.2%
3	10	0	2	Error text	Default_SupervisionFB	24	LAD_DB						
4													

Änderungen im Verhalten der Methoden ExportInstanceTextsToXlsx und ExportToXlsx in PlcAlarmTextProvider

Vor TIA Portal Openness V17 wird die Datei beim Exportieren überschrieben, wenn sie (als Pfadparameter) bereits existiert. Ab TIA Portal Openness V17 ändert sich dies und dem Benutzer wird eine Ausnahme UserException ausgelöst, wenn die Datei (als Pfadparameter) bereits existiert.

Änderungen im Verhalten der Export API für die Übernahme eines spezifischen Pfads

Vor TIA Portal Openness V17 sind spezifische Pfade in der Export API nicht erlaubt. Wenn ein spezifischer Pfad angegeben wird, wird die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException ausgelöst.

Ab TIA Portal Openness V17 ändert sich dies und spezifische Pfade werden erlaubt.

In den folgenden Fällen erlaubt TIA Portal Openness API einen spezifischen Pfad.

Eingangspfad	TIA Portal Openness V17
D:\SampleDirectory\Sample	Es wird keine Ausnahme ausgelöst
D:\SampleDirectory\Sample	Es wird keine Ausnahme ausgelöst
D:\SampleDirectory\Sample	Es wird keine Ausnahme ausgelöst

Eingangspfad	TIA Portal Openness V17
D:\SampleDirectory\Sample	Es wird keine Ausnahme ausgelöst
D:\\SampleDirectory\Sample	Es wird keine Ausnahme ausgelöst
D:\SampleDirectory\\Sample	Es wird keine Ausnahme ausgelöst
D:\\\\SampleDirectory\\Sample	Es wird keine Ausnahme ausgelöst
D:\SampleDirectory\\\Sample	Es wird keine Ausnahme ausgelöst
\\SampleDirectory\Sample	Es wird keine Ausnahme ausgelöst
\\SampleDirectory\\Sample	Es wird keine Ausnahme ausgelöst
\\\\SampleDirectory\\Sample	Es wird keine Ausnahme ausgelöst
\\\\SampleDirectory\\Sample	Es wird keine Ausnahme ausgelöst
//SampleDirectory/Sample	Es wird keine Ausnahme ausgelöst
"SampleDirectory"	Die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException wird beim Exportieren von Openness API ausgelöst.
"//SampleDirectory"	Die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException wird beim Exportieren von Openness API ausgelöst.
"D:\SampleDirectory\Sample\..lotherSample"	Die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException wird beim Exportieren von Openness API ausgelöst.
"SampleDirectory\Sample"	Die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException wird beim Exportieren von Openness API ausgelöst.
"..\Sample", "SampleName"	Die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException wird beim Exportieren von Openness API ausgelöst.
"D:\ SampleDirectory\Sample"	Die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException wird beim Exportieren von Openness API ausgelöst.
D:\SampleDirectory\ \Sample	Die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException wird beim Exportieren von Openness API ausgelöst.
D:\SampleDirectory\Sample	Die Ausnahme EngineeringTargetInvocationDetailException.SpecificPathException wird beim Exportieren von Openness API ausgelöst.

Geänderter Datentyp des Attributs ScreenNumber

Ab der TIA Portal Openness V17 Siemens.Engineering.dll ändert sich der Datentyp des Attributs ScreenNumber von Byte nach UInt16.

Geändertes Verhalten beim CAX-Export/Import

Ab TIA Portal Openness V17 werden die folgenden Änderungen für den CAX-Export/Import vorgenommen:

- Unterstützung für benutzerspezifische Attribute auf GSD/GSDML-Geräten.
- Ein Benutzer kann CAX-Daten in Multiuser-Projekten und UMAC-geschützten Projekten austauschen.

- Unterstützung beim Austausch von CAX-Daten mit normiertem Typenidentifizierer. Diese Funktion wird über die CAX-Benutzeroberfläche gesteuert.
- Die Struktur des PN-PN-Kopplers in einer AML-Datei wurde verändert. Damit sollen interne Strukturunterschiede in den am Austausch beteiligten Tools (TIAP und ECAD-Systeme) verhindert und eine gemeinsame Struktur für den Austausch bereitgestellt werden.
- Der Export/Import von Plus PC-Stationen wird unterstützt. Anmerkung: Der Austausch mit EPLAN- und anderen Tools funktioniert nicht, da die definierten zusätzlichen Attribute in AR APC nicht definiert sind.
- Beim Importieren erfolgt eine Benachrichtigung des Benutzers über die Neuzuweisung der Startadresse/Netzwerkadresse, wenn diese sich durch interne Vorgänge in TIA Portal geändert hat. Beispiel: Ausfälle von Subnetz-/Peripheriesystemverbindungen, überlappende Adressen usw. Diese Funktion wird über die Einstellungen der CAX-Benutzeroberfläche gesteuert.
- Benachrichtigung mit einer Warnmeldung für den Benutzer, wenn dieser eine AML-Datei mit OMS und PLCs mit aktivierter Sicherheit importiert.
- Ein neues Attribut 'ProfinetDeviceName' für alle Ethernet-Teilnehmer wird eingeführt.
- Ein Gerät mit steckbarem Profinet/Profibus Schnittstellen-Geräteelement wird immer mit getrennten Rollen exportiert. Ein steckbares Schnittstellen-Geräteelement im TIA-Projekt wird als steckbares Geräteelement mit der Rolle DeviceItem exportiert (mit einem 'neuen' untergeordneten Geräteelement mit der Rolle CommunicationInterface). Der Import älterer AML-Dateien muss jedoch immer unterstützt werden.
- Unterstützung eines toleranteren Imports der Startadresse (unabhängig von der IoType-Reihenfolge in DeviceItem).
- Der Import kompakter Module mit Hilfe von 'TemplateReference' wird mit einigen Einschränkungen unterstützt.

Unterstützung für Bausteine der neuen Programmiersprache CEM

Mit TIA Portal Openness V17 wurde die neue Programmiersprache CEM (Cause Effect Matrix) eingeführt. Wenn Sie auf eine Instanz eines solchen Bausteins treffen, löst der Zugriff auf das Attribut zur Programmiersprache in jeder Siemens.Engineering.dll < V17 eine Ausnahme aus. Bei Engineering.dll V17 wird die richtige Programmiersprache ausgegeben. Bausteine der Programmiersprache CEM unterstützen nicht die Exportmethode irgendeiner Siemens.Engineering.dll.

Unterstützung der neuen Bibliothekstypen "Unified Faceplate" und "Unified Graphic"

Mit TIA Portal Openness V17 wurden die neuen Bibliothekstypen "Unified Faceplate" und "Unified Graphic" eingeführt. Diese Typen werden in Openness nicht unterstützt. Wenn Sie auf Instanzen dieser Typen treffen, lösen alle Siemens.Engineering.dll eine Ausnahme aus.

7.3 Größere Änderungen in TIA Portal Openness V16

Änderungen

Wenn Sie die Hinweise zur versionsübergreifenden Programmierung beachtet haben und Ihr Projekt nicht auf V16 aktualisieren, laufen Ihre Anwendungen ohne jede Einschränkung auf jedem Rechner, auch wenn nur ein TIA Portal V16 installiert ist.

Wenn Sie Ihr Projekt auf V16 aktualisieren, ist es notwendig, Ihre Anwendung mit der SiemensEngineering.dll von V16 neu zu übersetzen. In manchen Fällen kann es erforderlich sein, den Code Ihrer Anwendung anzupassen.

Exportieren von Datenbausteinen mit Export-Option "Keine"

Ab TIA Portal Openness V16 werden schreibgeschützte Teile eines Datenbausteins als informative Elemente exportiert. Diese Attribute können in der exportierten XML nicht mehr geändert werden.

Adressattribute von I&M und Profisafe für Drucktaster- und Tastentafeln

Ab TIA Portal Openness V16 stehen Adressattribute von I&M und Profisafe für die Drucktaster- und Tastentafeln auf Modulebene zur Verfügung.

Exportieren von Datenbausteinattributen der Eigenschaft "Speicheraufbau"

Ab TIA Portal Openness V16 werden Instanz-DBs von FBs, Attribute von ARRAY-DBs und GRAPH-Bausteinen mit ReadOnly="True" exportiert.

Attribut Access in Einheiten

Ab TIA Portal Openness V16 wird der nicht konsistente Import und Export des Attributs "Access" in Bausteinen, UDTs und der Variablentabelle in Einheiten unterstützt

Schemadefinition der XML-Datei von SimaticML

Ab TIA Portal Openness V16 ändert sich der Standardwert des Attributs SystemDefined für den Teilnehmer des booleschen Attributs in der Schemadefinition der XML-Datei von SimaticML zu "False". Diese Änderung hat jedoch keine Auswirkungen auf Funktionen für den Export/Import von XML-Dateien. Sie ist nur relevant für Benutzer, die versuchen, XML-Dateien mit Hilfe der Schemadefinition zu generieren.

XML-Importmethode für Bausteine und Typen verbessert

Ab TIA Portal Openness V16 verfügt die neue Importmethode über zusätzliche Parameter (path, importOptions und swImportOptions) sowie den neuen Enumerationswert IgnoreUnitAttribute. Mit swImportOptions.IgnoreUnitAttributes können Sie einen Baustein aus einer Einheit in eine Zusammenstellung von Nichteinheiten importieren.

AML-GUIDs werden in App-IDs gespeichert

Ab TIA Portal Openness V16 werden AML-GUIDs im Attribut CustomIdentity (App-IDs) und nicht mehr in Comment gespeichert, um den Round-Trip-Austausch von Gerät und Modul zu unterstützen.

Name von HMI-Objektklassen

Ab TIA Portal Openness V16 enthält die folgende Tabelle eine Liste der erforderlichen Namensänderungen für HMI-Objektklassen:

Klassenname	Neuer Klassenname
AnalogAlarmComposition	HmiAnalogAlarmComposition
DiscreteAlarmComposition	HmiDiscreteAlarmComposition
AlarmClassComposition	HmiAlarmClassComposition
DataLogComposition	HmiDataLogComposition
AlarmLogComposition	HmiAlarmLogComposition
LoggingTagComposition	HmiLoggingTagComposition
LogConfiguration	DataLog

Eigenschaftensname für Datenarchiv/Meldearchiv hinzugefügt/entfernt

Ab TIA Portal Openness V16 werden die folgenden neuen Eigenschaften "StorageDevice" und "StorageFolder" dem DataLog/AlarmLog (Datenarchiv/Meldearchiv) hinzugefügt und Eigenschaften wie "StoragePath" und "RequireExplicitRelease" werden aus DataLog/AlarmLog und ScreenItems entfernt.

Das aktualisierte Codebeispiel für die Eigenschaften StorageDevice und StorageFolder finden Sie hier:

```
HmiDataLog dataLog = hmiSoftware.DataLogs.Find("DataLog1");
dataLog.Settings.StorageDevice = DeviceNode.Local;
dataLog.Settings.StorageFolder = @"D:\workdir\DataLogs";
```

Eigenschaftensname aus HMI Tag entfernt

Ab TIA Portal Openness V16 ist die Eigenschaft 'DisplayName' aus dem Objekt HMI Tag entfernt.

7.4 Größere Änderungen in TIA Portal Openness V15.1

Änderungen

Wenn Sie die Hinweise zur versionsübergreifenden Programmierung beachtet haben und Ihr Projekt nicht auf V15.1 aktualisieren, laufen Ihre Anwendungen ohne jede Einschränkung auf jedem Rechner, selbst wenn nur ein TIA Portal V15.1 installiert ist.

Wenn Sie Ihr Projekt auf V15.1 aktualisieren, ist es notwendig, Ihre Anwendung mit der SiemensEngineering.dll von V15.1 neu zu übersetzen. In manchen Fällen kann es erforderlich sein, den Code Ihrer Anwendung anzupassen.

Typkennungen

Die Typkennung für Baugruppenträger und Geräte der Art "PC mit Ports" sowie "Ethernet-Gerät mit Ports" wurde geändert.

PC mit Ports	Typkennung vor V15.1	Typkennung ab V15.1
Gerät (device)	System:DesktopPC.Device	System:Device.DesktopPC
Baugruppenträger (rack)	System:DesktopPC.Rack	System:Rack.DesktopPC
Geräteelement	System:DesktopPC.Port<X> <X> denotes the number of ports	System:DeviceItem.EthernetDevice.Port<X> <X> denotes the number of ports

Ethernet-Geräte mit Ports		
Gerät (device)	System:DummyPC.Device	System:Device.EthernetDevice
Baugruppenträger (rack)	System:Rack.DummyPC	System:Rack.EthernetDevice
Geräteelement	System:DummyPC.Port<X> <X> denotes the number of ports	

Ausfälle beim Versuch, mit dem TIA Portal zu verbinden

Führt der Versuch, eine Verbindung zum TIA Portal herzustellen, zu einem Ausfall, ist die daraufhin angezeigte Meldung nun spezifischer.

Thread-übergreifender Betrieb

Der Zugriff auf Openness-Objekte ist nicht inhärent threadsicher. Wenn Sie Multithreading zur Verbesserung der Performance Ihrer Openness-Anwendung einsetzen, wird die Erstellung der Instanz des TIA Portals mit MTA empfohlen. Wenn TIA Portal innerhalb eines STA-Threads erstellt oder angehängt wird, muss auf alle Openness-Objekte in Zusammenhang mit dieser Instanz des TIA Portals aus demselben STA-Thread zugegriffen werden; andernfalls wird eine Ausnahme ausgelöst.

Submodule haben nicht die Attribute Author und TypeName.

Die Attribute Author und TypeName wurden von Submodulen, die nicht gesteckt werden können, entfernt.

Öffnen einer globalen Bibliothek

Ab TIA Portal Openness V15.1 kann eine über Openness, unabhängig von einem bestehenden Vorschaumodus der Bibliothek, eine globale Bibliothek geöffnet werden.

Code bei Anwendungsabbruch

Im Fall eines Codes bei Anwendungsabbruch geschieht Folgendes:

- Bis zu TIA Portal Openness V15 erhalten Sie einen Ausnahmefehler ohne Wiederherstellbarkeit
- Ab TIA Portal Openness V15.1 erhalten Sie eine `EngineeringRuntimeException` oder eine `EngineeringTargetInvocationException`, wenn der Fehlercode bekannt ist, oder einen Ausnahmefehler ohne Wiederherstellbarkeit, wenn der Fehlercode unbekannt ist.

Schemaerweiterung für namenlose Parameter

SCL-Bausteine können importiert werden, selbst wenn ENO bei einem nicht formellen Aufruf genutzt wird.

Header indizierter Parameter

Ab TIA Portal Openness V15.1 kann der Header indizierter Parameter nicht über Openness geändert werden.

Einige Antriebsparameter werden als indizierte Parameter modelliert, da sie mehrere Datenstücke zu einem Thema bieten. Die Modellierung der indizierten Antriebsparameter in Openness folgt der antriebsspezifischen Definition der jeweiligen Parameter wie im entsprechenden Listenhandbuch festgelegt.

Indizierte Parameter werden wie folgt modelliert:

Header-Element: Der jeweilige Antriebsparameter ohne Index. Das Header-Element enthält einen beschreibenden Text, der Sie über die Semantik des referenzierten indizierten Antriebsparameters informiert. Deshalb ist das Header-Element schreibgeschützt, weil es keinen Istwert enthält.

Indizierte Elemente: Die indizierten Elemente des Antriebsparameters unter dem Header-Element. Sie liefern beschreibenden Text, der die Semantik des spezifischen indizierten Elements definiert (schreibgeschützt). Zusätzlich geben sie auch den Wert an, der über die Openness API abgerufen werden kann. Wenn der jeweilige indizierte Parameter auch beschreibbar ist, kann der Wert auch über die Openness API festgelegt werden.

Attribut TransmissionRateAndDuplex

Einige fehlerhafte Enumerationswerte für das Attribut `TransmissionRateAndDuplex` wurden korrigiert, beispielsweise wurde der Enumerationswert `"POFPCF100MbpsFullDuplexLD"` entfernt und `"POFPCF100MbpsFullDuplex"` wurde hinzugefügt. Ausführliche Informationen finden Sie unter Konfigurierbare Attribute einer Port-zu-Port-Verbindung (Seite 201).

Attribut AutoNumber für Bausteine mit Know-how-Schutz

Ab V15.1 kann das Attribut `AutoNumber` nicht über TIA Portal Openness geändert werden, wenn ein Baustein über Know-how-Schutz verfügt.

Anzahl der von Schnittstelle ChannelInfo gelisteten Kanäle

Bis TIA Portal Openness V15 zeigt die Schnittstelle ChannelInfo die Anzahl der verfügbaren Kanälen bei manchen Modulen nicht richtig an.

Zugriff auf ProDiag-FB-Attribute

Auf die folgenden Attribute eines ProDiag-Funktionsbausteins kann über TIA Portal Openness zugegriffen werden:

- Version
- Erstwerterfassung
- Nutzung zentraler Zeitstempel

Import/Export fehlersicherer Bausteine

Der Import fehlersicherer Bausteine aus früheren Versionen ist nicht möglich.

Der Export von systemgenerierten fehlersicheren Bausteinen wird ab TIA Portal Openness V15.1 verhindert.

R/H-Systeme

Der Download bzw. Zugriff für R/H-Geräte ist auf dem Gerät verfügbar.

Online- und Download-Provider sind für einzelne R/H PLCs (Deviceltems) nicht verfügbar.

Für PLC2 eines R/H-Systems ist SoftwareContainer nicht verfügbar.

7.5 Größere Änderungen in TIA Portal Openness V15

Änderungen

Wenn Sie die Hinweise zur versionsübergreifenden Programmierung beachtet haben und Ihr Projekt nicht auf V15 aktualisieren, laufen Ihre Anwendungen ohne jede Einschränkung auf jedem Rechner, selbst wenn nur ein TIA Portal V15 installiert ist.

Wenn Sie Ihr Projekt auf V15 aktualisieren, ist es notwendig, Ihre Anwendung mit der SiemensEngineering.dll von V15 neu zu übersetzen.

In manchen Fällen ist es notwendig, den Code Ihrer Anwendung anzupassen

- Verhaltensänderungen für Zusammensetzungen in DeviceltemComposition
- BitOffset von ASi-Adressen
- Klasse Exception
- Systemordner von System-UDTs
- Submodule haben nicht die Attribute Author und TypeName
- Zeitstempel für letzte Änderung

- Export-XML für GRAPH-Bausteine
- Variablentabellen importieren
- Nicht fehlersichere relevante Attribute eines PLC ändern
- F-Parameter bei eingestelltem Sicherheitspasswort ändern
- Zugriff auf Objekte in einer S7-1200 CPU

Verhaltensänderungen für Zusammensetzungen in DeviceItemComposition

Die folgenden Zusammensetzungen in DeviceItemComposition wurden geändert, um ein dynamisches Verhalten zu erreichen: Die Zusammensetzung wird jetzt aktualisiert, wenn ein Element über die Benutzeroberfläche des TIA Portals hinzugefügt oder gelöscht wird.

- IoSystem – ConnectedIoDevices
- Subnet – IoSystems
- Subnet – Nodes
- NetworkInterface – Nodes
- NetworkInterface – Ports
- NetworkPort – ConnectedPorts
- SubnetOwner – Subnets

BitOffset von ASi-Adressen

Wenn ein Modul eine Eingangs- und eine Ausgangsadresse für beide Adressobjekte hat, wird das korrekte Attribut BitOffset bereitgestellt.

Wenn ein Modul Kanäle hat, wird das Attribut BitOffset für den Kanal nicht bereitgestellt.

Klasse Exception

ServiceID und MessageID wurden aus der Klasse exception entfernt.

Submodule haben nicht die Attribute Author und TypeName.

Die Attribute Author und TypeName wurden von Submodulen, die nicht gesteckt werden können, entfernt.

Systemordner von System-UDTs

Bei Systemordnern von System-UDTs werden der entsprechende Ordner und die entsprechende Zusammensetzung bereitgestellt. Das führt auch zu einer Änderung in der Hierarchie von Vergleichsergebnissen.

Zeitstempel für letzte Änderung

Wenn während eines Upgrades ein Objekt geändert wird, ändert sich auch der Zeitstempel für die letzte Änderung.

Export-XML für GRAPH-Bausteine

Die Export-XML für GRAPH-Bausteine enthält eine zusätzliche leere Aktion: `<Actions />`.

Variablentabellen importieren

Die Einstellung von Variablenattributen ist nicht mehr von Datentypen abhängig.

Nicht fehlersichere relevante Attribute eines PLC ändern

Alle nicht fehlersicheren relevanten Attribute eines PLC können über TIA Portal Openness geändert werden, selbst wenn ein Sicherheitspasswort eingestellt ist.

F-Parameter bei eingestelltem Sicherheitspasswort ändern

F-Parameter von F-IO können nur geändert werden, wenn das Sicherheitspasswort nicht eingestellt ist.

Zugriff auf Objekte in einer S7-1200 CPU

Der Zugriff auf Array-Variablen für die Technologieobjekte `TO_PositioningAxis` und `TO_CommandTable` wurde geändert. Einzelheiten hierzu finden Sie im Kapitel über S7-1200 Motion Control.

7.6 Die wichtigsten Änderungen in V14 SP1

7.6.1 Die wichtigsten Änderungen in V14 SP1

Einleitung

Die folgenden Änderungen wurden bei dem TIA Portal Openness API-Objektmodell V14 SP1 vorgenommen, was möglicherweise Auswirkungen auf Ihre vorhandenen Anwendungen hat:

Änderung	Erforderliche Anpassung von Programmcode
Verbesserte Handhabung von Masterkopien	Die Aktion CreateFrom erstellt ein neues Objekt basierend auf einer Masterkopie in einer Bibliothek und platziert es in der Zusammensetzung, wo die Aktion aufgerufen wurde. Die Aktion CreateFrom unterstützt nur Masterkopien, die ausschließlich einzelne Objekte enthalten. Der Ausgabetypp entspricht dem jeweils zusammengesetzten Typ. Die folgenden Zusammensetzungen unterstützen CreateFrom: • Siemens.Engineering.HW.DeviceComposition • Siemens.Engineering.HW.DeviceItemComposition • Siemens.Engineering.SW.Blocks.PlcBlockComposition • Siemens.Engineering.SW.Tags.PlcTagTableComposition • Siemens.Engineering.SW.Tags.PlcTagComposition • Siemens.Engineering.SW.Types.PlcTypeComposition • Siemens.Engineering.SW.TechnologicalObjects.TechnologicalInstanceDBComposition • Siemens.Engineering.SW.Tags.PlcUserConstantComposition • Siemens.Engineering.Hmi.Tag.TagTableComposition • Siemens.Engineering.Hmi.Tag.TagComposition • Siemens.Engineering.Hmi.Screen.ScreenComposition • Siemens.Engineering.Hmi.Screen.ScreenTemplateComposition • Siemens.Engineering.Hmi.RuntimeScripting.VBScriptComposition • Siemens.Engineering.HW.SubnetComposition • Siemens.Engineering.HW.DeviceUserGroupComposition • Siemens.Engineering.SW.Blocks.PlcBlockUserGroupComposition • Siemens.Engineering.SW.ExternalSources.PlcExternalSourceUserGroupComposition • Siemens.Engineering.SW.Tags.PlcTagTableUserGroupComposition • Siemens.Engineering.SW.Types.PlcTypeUserGroupComposition
Verbesserte Handhabung globaler Bibliotheken	Bestehende Aktionen mit globalen Bibliotheken können jetzt Änderungsaktionen umfassen, z. B. Löschen einer Masterkopie aus einer globalen Bibliothek. UpdateProject und UpdateLibrary verwenden nicht länger die Parameter UpdatePathsMode und DeleteUnusedVersionsMode. Nicht verwendete Versionen werden nach einem Update nicht gelöscht

Änderung	Erforderliche Anpassung von Programmcode
Änderung von System.String in System.IO.FileInfo Änderung von System.String in System.IO.DirectoryInfo	Alle Ereignisse, bei denen ein String-Pfad spezifiziert werden muss, verwenden den Pfad FileInfo oder den Pfad DirectoryInfo. Beispiel: • Projekt öffnen • Bibliothek öffnen • Projekt erstellen • Globale Bibliothek erstellen • ...

Neue Elemente im Objektmodell

Name	Typ	Namensraum	Kommentar
PlcUserConstant	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstant
PlcUserConstantComposition	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstantComposition
PlcSystemConstant	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstant
PlcSystemConstantComposition	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstantComposition
MultilingualTextItem	Klasse	Siemens.Engineering	Auf mehrsprachige Texte zugreifen.
MultilingualTextItemComposition	Klasse	Siemens.Engineering	Auf mehrsprachige Texte zugreifen.
TiaPortalTrustAuthority.Feature-Tokens	Enumerationswert	Siemens.Engineering	Auf Einstellungen des TIA Portals zugreifen.
TiaPortalSetting	Klasse	Siemens.Engineering.Settings	Auf Einstellungen des TIA Portals zugreifen.
TiaPortalSettingComposition	Klasse	Siemens.Engineering.Settings	Auf Einstellungen des TIA Portals zugreifen.
TiaPortalSettingsFolder	Klasse	Siemens.Engineering.Settings	Auf Einstellungen des TIA Portals zugreifen.
TiaPortalSettingsFolderComposition	Klasse	Siemens.Engineering.Settings	Auf Einstellungen des TIA Portals zugreifen.
LanguageAssociation	Klasse	Siemens.Engineering	Auf aktive Sprachen zugreifen.
LanguageComposition.Find	Methode	Siemens.Engineering	Auf aktive Sprachen zugreifen.

Geänderte Elemente im Objektmodell

Name	Typ	Namensraum	Kommentar
PlcConstant	Klasse	Siemens.Engineering.SW.Tags	Herausgegebene Basisklasse von PlcUserConstant und PlcSystemConstant.
PlcTag	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstantComposition
ITargetComparable	Schnittstelle	Siemens.Engineering.Compare	String-Attribut DataTypeName anstelle eines offenen Links DataType.
MultilingualText	Klasse	Siemens.Engineering	Auf mehrsprachige Texte zugreifen.

Name	Typ	Namensraum	Kommentar
ProjectComposition.Create	Methode	Siemens.Engineering	Parameter geändert zur Verwendung von DirectoryInfo und eines Strings.
Project.Subnets	Attribut	Siemens.Engineering	Auf Subnetze zugreifen
Project.Languages	Attribut	Siemens.Engineering	Verschieben, um Attribut von Siemens.Engineering.LanguageSettings darzustellen und unterstützte Sprachen bereitzustellen

Gelöschte Elemente im Objektmodell

Name	Typ	Namensraum	Kommentar
PlcConstantComposition	Klasse	Siemens.Engineering.SW.Tags	Teilen in PlcSystemConstantComposition und PlcUserConstantComposition.
CompareResultElement.PathInformation	Attribut	Siemens.Engineering.SW.Tags	Nicht mehr verwendet.
MultilingualText.GetText(CultureInfo cultureInfo)	Methode	Siemens.Engineering.Compare	Geändertes Konzept für den Zugriff auf Textelemente von MultilingualText.
TiaPortalTrustAuthority.CustomerIdentification	Enumerationswert	Siemens.Engineering	Nicht mehr verwendet.
TiaPortalTrustAuthority.ElevatedAccessExtensions	Enumerationswert	Siemens.Engineering	Nicht mehr verwendet.

Verhaltensänderungen

Name	Typ	Namensraum	Kommentar
PlcTag.Export(FileInfo path, ExportOptions options)	Methode	Siemens.Engineering.SW.Tags	Der Wert des Attributs LogicalAddress wird jetzt immer in internationaler Mnemonik exportiert. Deutsche Mnemonik wird für den Import noch akzeptiert.
PlcTag.LogicalAddress	Attribut	Siemens.Engineering.SW.Tags	Der Wert des Attributs LogicalAddress wird jetzt immer in internationaler Mnemonik zurückgegeben. Deutsche Mnemonik wird für Schreiben noch akzeptiert.

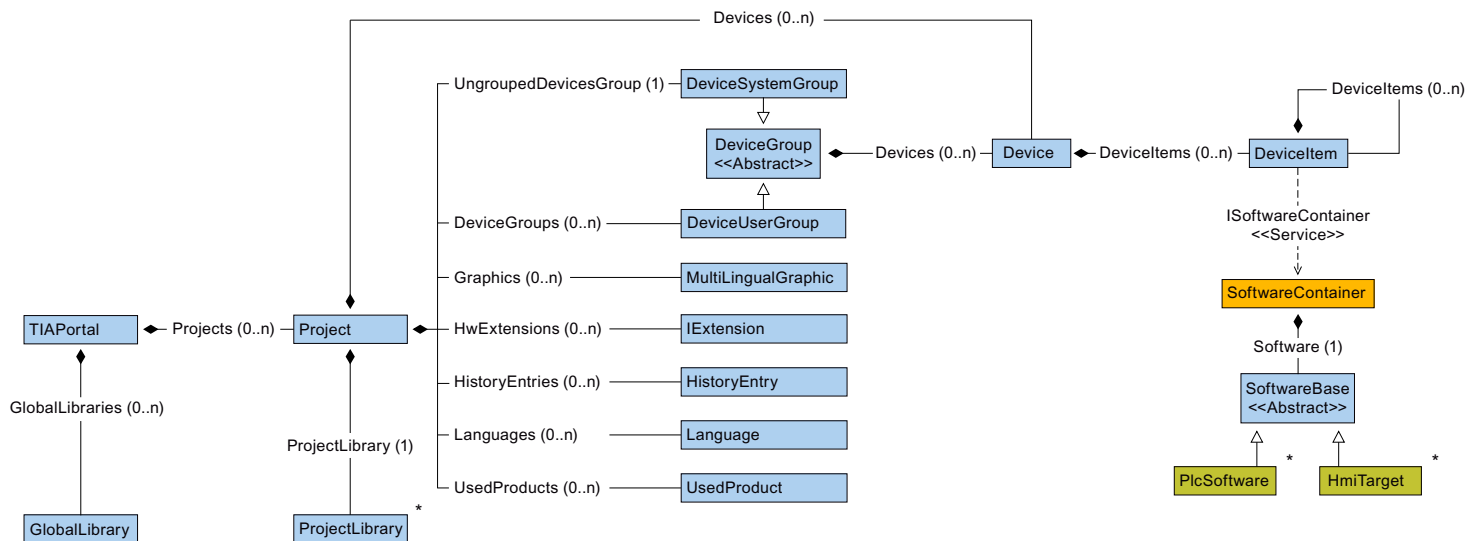
7.6.2 Wesentliche Änderungen im Objektmodell

Objektmodell von TIA Portal Openness V14

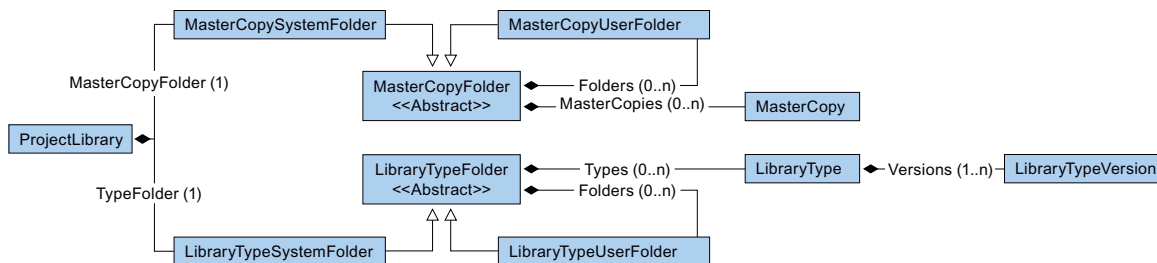
Um Ihnen einen Vergleich zwischen dem alten und dem neuen Objektmodell von TIA Portal Openness zu ermöglichen, beschreibt das nachstehende Diagramm das Objektmodell von TIA Portal V14.

Hinweis

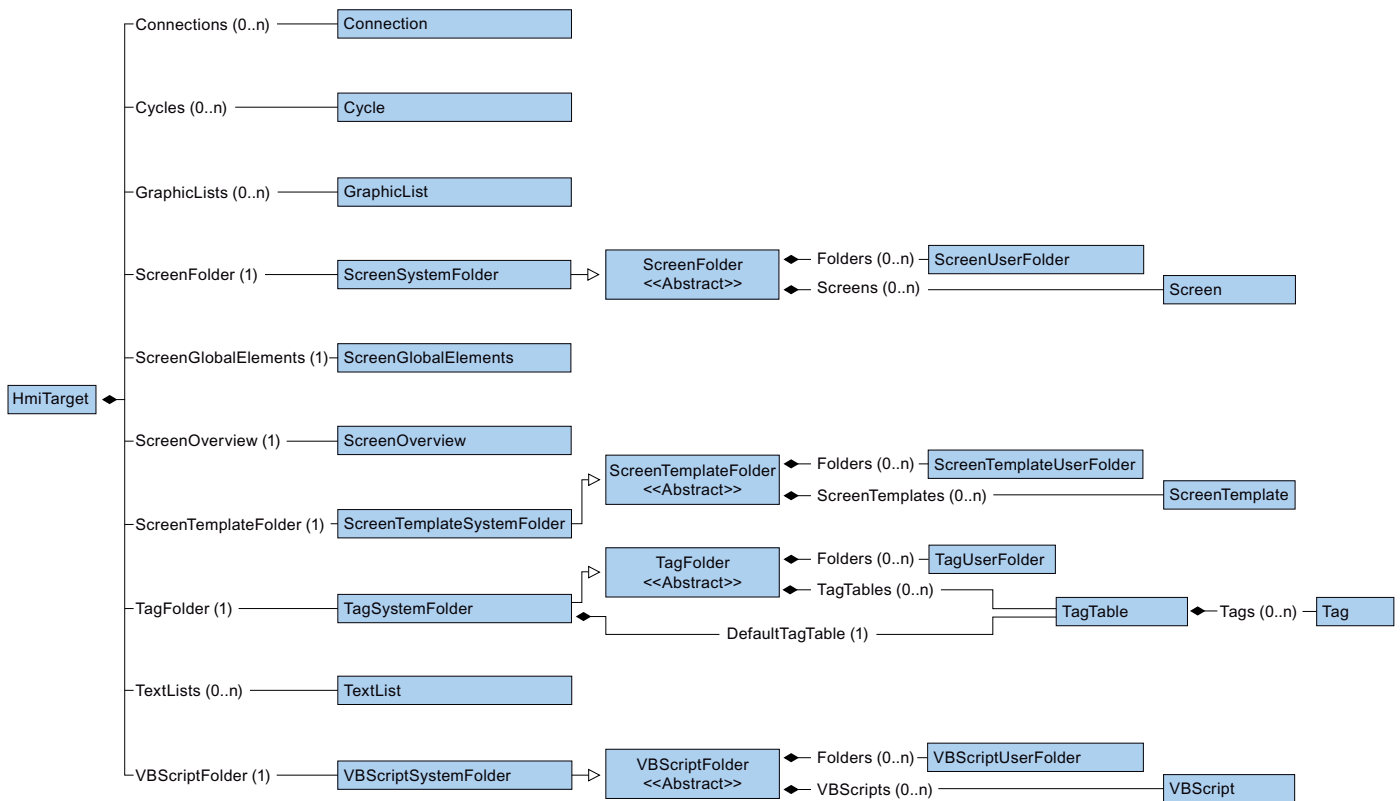
Das in dem Diagramm beschriebene Objektmodell ist veraltet. Sie finden Informationen über das Objektmodell von TIA Portal Openness V14 SP1 unter TIA Portal Openness-Objektmodell (Seite 55)



Das folgende Diagramm beschreibt die Objekte, die sich unter ProjectLibrary befinden.



Das folgende Diagramm beschreibt die Objekte, die sich unter HmiTarget befinden.

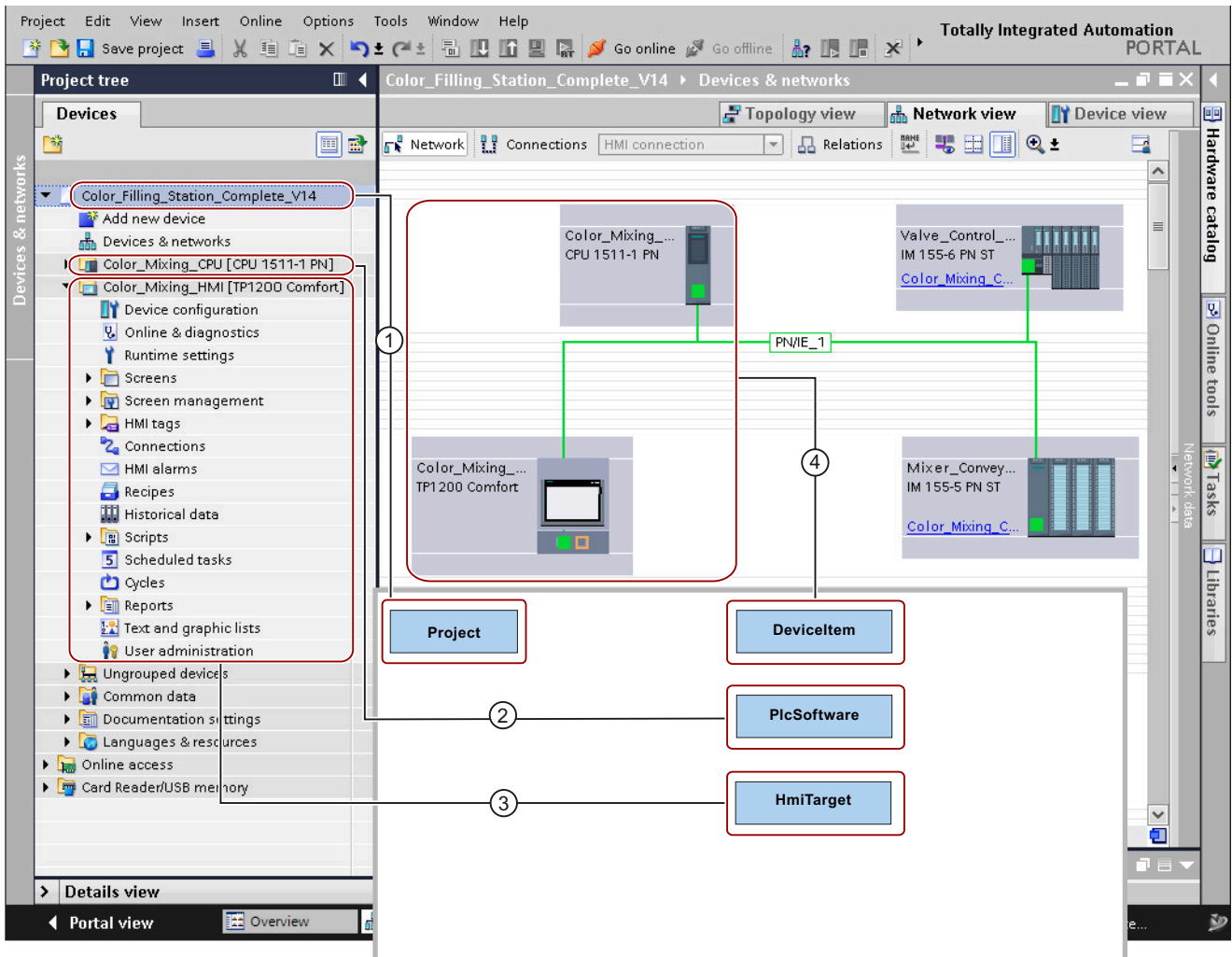


Das folgende Diagramm beschreibt die Objekte, die sich unter PlcSoftware befinden.



Beziehung zwischen TIA Portal und TIA Portal Openness-Objektmodell

Das Bild unten zeigt die Beziehung zwischen dem Objektmodell und einem Projekt im TIA Portal:



- ① Das Objekt "Project" entspricht einem offenen Projekt im TIA Portal.
- ② Das Objekt "PlcSoftware" ist vom Typ "SoftwareBase"④ und entspricht einem PLC. Der Inhalt des Objekts entspricht einem PLC in der Projektnavigation mit Zugriff auf Objekte wie Bausteine oder PLC-Variablen.
- ③ Das Objekt "HmiTarget" ist vom Typ "SoftwareBase"④ und entspricht einem Bediengerät. Der Inhalt des Objekts entspricht einem HMI-Gerät in der Projektnavigation mit Zugriff auf Objekte wie Bilder oder HMI-Variablen.
- ④ Das Objekt "Deviceltem" entspricht einem Objekt im Editor "Geräte & Netze". Ein Objekt vom Typ "Deviceltem" kann ein Baugruppenträger oder ein Einschubmodul sein.

7.6.3 Änderungen an der Pilotfunktionalität

Einleitung

Die folgenden Änderungen wurden bei dem API-Objektmodell V14 SP1 vorgenommen und sind nur relevant für Anwender, die die Pilotfunktionalität von HW Konfig in V14 genutzt haben:

Änderungen an TIA Portal Openness API-Typen

TIA Portal Openness API-Typ	Neuer TIA Portal Openness API-Typ
Siemens.Engineering.HW.IAddress	Siemens.Engineering.HW.Address
Siemens.Engineering.HW.IAddressController	Siemens.Engineering.HW.Features.AddressController
Siemens.Engineering.HW.IChannel	Siemens.Engineering.HW.Channel
Siemens.Engineering.HW.IDevice	Siemens.Engineering.HW.Device
Siemens.Engineering.HW.IDeviceItem	Siemens.Engineering.HW.DeviceItem
Siemens.Engineering.HW.IExtension	Siemens.Engineering.HW.Extensions
Siemens.Engineering.HW.IGsd	Siemens.Engineering.HW.Features.GsdObject
Siemens.Engineering.HW.IGsdDevice	Siemens.Engineering.HW.Features.GsdDevice
Siemens.Engineering.HW.IGsdDeviceItem	Siemens.Engineering.HW.Features.GsdDeviceItem
Siemens.Engineering.HW.IHardwareObject	Siemens.Engineering.HW.HardwareObject
Siemens.Engineering.HW.IHwIdentifier	Siemens.Engineering.HW.HwIdentifier
Siemens.Engineering.HW.IHwIdentifierController	Siemens.Engineering.HW.Features.HwIdentifierController
Siemens.Engineering.HW.IIoConnector	Siemens.Engineering.HW.IoConnector
Siemens.Engineering.HW.IIoController	Siemens.Engineering.HW.IoController
Siemens.Engineering.HW.IIoSystem	Siemens.Engineering.HW.IoSystem
Siemens.Engineering.HW.IInterface	Siemens.Engineering.HW.Features.NetworkInterface
Siemens.Engineering.HW.Extensions.ModuleInformationProvider	Siemens.Engineering.HW.Utilities.ModuleInformationProvider
Siemens.Engineering.HW.INode	Siemens.Engineering.HW.Node
Siemens.Engineering.HW.OPCUAExportProvider	Siemens.Engineering.HW.Utilities.OpcUaExportProvider
Siemens.Engineering.HW.IPort	Siemens.Engineering.HW.Features.NetworkPort
Siemens.Engineering.HW.IRole	Siemens.Engineering.HW.Features.HardwareFeature
	Siemens.Engineering.HW.Features.DeviceFeature
	Siemens.Engineering.HW.Utilities.ModuleInformationProvider
Siemens.Engineering.HW.SoftwareBase	Siemens.Engineering.HW.Software
Siemens.Engineering.HW.ISubnet	Siemens.Engineering.HW.Subnet
Siemens.Engineering.HW.ISoftwareContainer	Siemens.Engineering.HW.Features.SoftwareContainer
Siemens.Engineering.HW.ISubnetOwner	Siemens.Engineering.HW.Features.SubnetOwner

Änderungen für Enums

TIA Portal Openness API-Typ	Daten- typ	Neuer TIA Portal Openness API-Typ	Daten- typ
Siemens.Engineering.HW.Enums.AddressContext		Siemens.Engineering.HW.AddressContext	
Siemens.Engineering.HW.Enums.AddressIoType		Siemens.Engineering.HW.AddressIoType	
Siemens.Engineering.HW.Enums.AttachmentType		Siemens.Engineering.HW.MediumAttachmentType	
Siemens.Engineering.HW.Enums.BaudRate		Siemens.Engineering.HW.BaudRate	
Siemens.Engineering.HW.Enums.BusLoad		Siemens.Engineering.HW.CommunicationLoad	
Siemens.Engineering.HW.Enums.BusProfile		Siemens.Engineering.HW.BusProfile	
Siemens.Engineering.HW.Enums.CableLength		Siemens.Engineering.HW.CableLength	
Siemens.Engineering.HW.Enums.CableName	Ulong	Siemens.Engineering.HW.CableName	Long
Siemens.Engineering.HW.Enums.ChannelloType	Byte	Siemens.Engineering.HW.ChannelloType	Int
Siemens.Engineering.HW.Enums.ChannelType	Byte	Siemens.Engineering.HW.ChannelType	Int
Siemens.Engineering.HW.Enums.DeviceItemClassifications		Siemens.Engineering.HW.DeviceItemClassifications	
Siemens.Engineering.HW.Enums.InterfaceOperatingModes		Siemens.Engineering.HW.InterfaceOperatingModes	
Siemens.Engineering.HW.Enums.IpProtocolSelection		Siemens.Engineering.HW.IpProtocolSelection	
Siemens.Engineering.HW.Enums.MediaRedundancyRole		Siemens.Engineering.HW.MediaRedundancyRole	
Siemens.Engineering.HW.Enums.NetType		Siemens.Engineering.HW.NetType	
Siemens.Engineering.HW.Enums.ProfinetUpdateTimeMode		entfernt	
Siemens.Engineering.HW.Enums.RtClass	Byte	Siemens.Engineering.HW.RtClass	Int
Siemens.Engineering.HW.Enums.SignalDelaySelection	Byte	Siemens.Engineering.HW.SignalDelaySelection	Int
Siemens.Engineering.HW.Enums.SyncRole	Byte	Siemens.Engineering.HW.SyncRole	Int
Siemens.Engineering.HW.Enums.TransmissionRateAndDuplex	UInt	Siemens.Engineering.HW.TransmissionRateAndDuplex	Int

Änderungen für Attributwerte von Siemens.Engineering.HW.IoConnector

Attribut	Datentyp	Neuer Name	Datentyp
ProfinetUpdateTimeMode	ProfinetUpdateTimeMode	PnUpdateTimeAutoCalculation	Bool
ProfinetUpdateTime		PnUpdateTime	
AdaptUpdateTime		PnUpdateTimeAdaption	
WatchdogFactor		PNWatchdogFactor	
		DeviceNumber	String

Änderungen für Attributwerte von Siemens.Engineering.HW.IoController

Attribut	Datentyp	Neuer Name	Datentyp
		DeviceNumber	String

Änderungen für Attributwerte von Siemens.Engineering.HW.Node

Attribut	Datentyp	Neuer Name	Datentyp
HighestAddress		entfernt, nur am Subnetz verfügbar	
TransmissionSpeed		entfernt, nur am Subnetz verfügbar	
IsoProtocolUsed		UseIsoProtocol	
IpProtocolUsed		UseIpProtocol	
RouterAddressUsed		UseRouter	
PnDeviceNameAutoGenerated		PnDeviceNameAutoGeneration	
DeviceNumber		entfernt, zu IO-Connector / IO-Controller verschoben	

Änderungen für Attributwerte von Siemens.Engineering.HW.Subnet

Attribut	Datentyp	Neuer Name	Datentyp
HighestAddress	Byte	HighestAddress	Int
CableConfiguration		PbCableConfiguration	
RepeaterCount		PbRepeaterCount	
CopperCableLength		PbCopperCableLength	
OpticalComponentCount		PbOpticalComponentCount	
OpticalCableLength		PbOpticalCableLength	
OpticalRingEnabled		PbOpticalRing	
Olmp12		PbOlmp12	
Olmg12		PbOlmg12	
Olmg12Eec		PbOlmg12Eec	
Olmg121300		PbOlmg121300	
AdditionalNetworkDevices		PbAdditionalNetworkDevices	
AdditionalDpMaster	Byte	PbAdditionalDpMaster	Int
TotalDpMaster	Byte	PbTotalDpMaster	Int
AdditionalPassiveDevice	Byte	PbAdditionalPassiveDevice	Int
TotalPassiveDevice	Byte	PbTotalPassiveDevice	Int
AdditionalActiveDevice	Byte	PbAdditionalActiveDevice	Int
TotalActiveDevice	Byte	PbTotalActiveDevice	Int
PbCommunicationLoad	BusLoad	PbAdditionalCommunicationLoad	CommunicationLoad
OptimizeDde		PbDirectDataExchange	
MinimizeTslot		PbMinimizeTslotForSlaveFailure	
OptimizeCableConfig		PbOptimizeCableConfiguration	
CyclicDistribution		PbCyclicDistribution	
TslotInit		PbTslotInit	
Tslot		PbTslot	
MinTsdr		PbMinTsdr	
MaxTsdr		PbMaxTsdr	
Tid1		PbTid1	
Tid2		PbTid2	

Attribut	Datentyp	Neuer Name	Datentyp
Trdy		PbTrdy	
Tset		PbTset	
Tqui		PbTqui	
Ttr		PbTtr	
TtrMs		entfernt	
TtrTypical		PbTtrTypical	
TtrTypicalMs		entfernt	
Watchdog		PbWatchdog	
WatchdogMs		entfernt	
Gap	Byte	PbGapFactor	Int
RetryLimit	Byte	PbRetryLimit	Int
IsochronMode		IsochronousMode	
AdditionalDevice		PbAdditionalPassivDeviceForIsochro- nousMode	
TotalDevice		PbTotalPassivDeviceForIsochronousMo- de	
DpCycleTimeAutoCalc		DpCycleMinTimeAutoCalculation	
TiToAutoCalc		IsochronousTiToAutoCalculation	
Ti		IsochronousTi	
To		IsochronousTo	

Änderungen für Attributwerte von Siemens.Engineering.Project

Attribut	Datentyp	Neuer Name	Datentyp
.HwExtensions		.HwUtilities	

Änderungen für Attributwerte von Siemens.Engineering.HW.Baudrate

Attribut	Datentyp	Neuer Name	Datentyp
BaudRate.BAUD_9600		BaudRate.BAUD9600	
BaudRate.BAUD_19200		BaudRate.BAUD19200	
BaudRate.BAUD_45450		BaudRate.BAUD45450	
BaudRate.BAUD_93750		BaudRate.BAUD93750	
BaudRate.BAUD_187500		BaudRate.BAUD187500	
BaudRate.BAUD_500000		BaudRate.BAUD500000	
BaudRate.BAUD_1500000		BaudRate.BAUD1500000	
BaudRate.BAUD_3000000		BaudRate.BAUD3000000	
BaudRate.BAUD_6000000		BaudRate.BAUD6000000	
BaudRate.BAUD_12000000		BaudRate.BAUD12000000	

Änderungen für Attributwerte von Siemens.Engineering.HW.CableLength

Attribut	Datentyp	Neuer Name	Datentyp
CableLength.Unknown		CableLength.None	
CableLength.Length_20m		CableLength.Length20m	
CableLength.Length_50m		CableLength.Length50m	
CableLength.Length_100m		CableLength.Length100m	
CableLength.Length_1000m		CableLength.Length1000m	
CableLength.Length_3000m		CableLength.Length3000m	

Änderungen für Attributwerte von Siemens.Engineering.HW.ChannelloType

Attribut	Datentyp	Neuer Name	Datentyp
ChannelloType.Unknown		ChannelloType.Complex	

Änderungen für Attributwerte von Siemens.Engineering.HW.IpProtocolSelection

Attribut	Datentyp	Neuer Name	Datentyp
IpProtocolSelection.Address-Tailoring		IpProtocolSelection.VialoController	

Änderungen für Attributwerte von Siemens.Engineering.HW.TransmissionRateAndDuplex

Attribut	Datentyp	Neuer Name	Datentyp
TransmissionRateAndDuplex.Unknown		TransmissionRateAndDuplex.None	
TransmissionRateAndDuplex.TP10Mbps_HalfDuplex		TransmissionRateAndDuplex.TP10MbpsHalfDuplex	
TransmissionRateAndDuplex.TP10Mbps_FullDuplex		TransmissionRateAndDuplex.TP10MbpsFullDuplex	
TransmissionRateAndDuplex.AsyncFiber10Mbps_HalfDuplex		TransmissionRateAndDuplex.AsyncFiber10MbpsHalfDuplex	
TransmissionRateAndDuplex.AsyncFiber10Mbps_FullDuplex		TransmissionRateAndDuplex.AsyncFiber10MbpsFullDuplex	
TransmissionRateAndDuplex.TP100Mbps_HalfDuplex		TransmissionRateAndDuplex.TP100MbpsHalfDuplex	
TransmissionRateAndDuplex.TP100Mbps_FullDuplex		TransmissionRateAndDuplex.TP100MbpsFullDuplex	
TransmissionRateAndDuplex.FO100Mbps_FullDuplex		TransmissionRateAndDuplex.FO100MbpsFullDuplex	
TransmissionRateAndDuplex.X1000Mbps_FullDuplex		TransmissionRateAndDuplex.X1000MbpsFullDuplex	
TransmissionRateAndDuplex.FO1000Mbps_FullDuplex_LD		TransmissionRateAndDuplex.FO1000MbpsFullDuplexLD	
TransmissionRateAndDuplex.FO1000Mbps_FullDuplex		TransmissionRateAndDuplex.FO1000MbpsFullDuplex	

Attribut	Daten- typ	Neuer Name	Daten- typ
TransmissionRateAndDuplex.TP1000Mbps_FullDuplex		TransmissionRateAndDuplex.TP1000MbpsFullDuplex	
TransmissionRateAndDuplex.FO10000Mbps_FullDuplex		TransmissionRateAndDuplex.FO10000MbpsFullDuplex	
TransmissionRateAndDuplex.FO100Mbps_FullDuplex_LD		TransmissionRateAndDuplex.FO100MbpsFullDuplexLD	
TransmissionRateAndDuplex.POFPCF100Mbps_FullDuplex_LD		TransmissionRateAndDuplex.POFPCF100MbpsFullDuplexLD	

7.6.4 Änderungen bei Export und Import

7.6.4.1 Änderungen bei Export und Import

Einleitung

Der Export und Import mittels TIA Portal Openness API wurde in V14 SP1 erweitert, um Kommentare bei Array-Elementen verarbeiten zu können. Dadurch wurde ein neues Schema erforderlich. Für den Import und Export von Bausteinschnittstellen werden ab jetzt zwei Schemaversionen verwendet:

- Für den Import: Die Entscheidung über die verwendete Schemaversion wird anhand des Namensraums getroffen: <Sections xmlns=http://www.siemens.com/automation/Openness/SW/Interface/v2>
- Für den Export: Die Entscheidung über die verwendete Schemaversion wird anhand der Projektversion getroffen. Projekte V14 SP1 führen zu Version 2, Projekte V14 führen zu Version 1.

7.6.4.2 Änderungen in der API

Quelle generieren

Die folgenden Methoden wurden aus ProgramBlocks entfernt:

- GenerateSourceFromBlocks
- GenerateSourceFromTypes

Die folgenden Methoden wurden ergänzt:

- GenerateSource bis PlcExternalSourceSystemGroup

Beispiel

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
using HmiTarget = Siemens.Engineering.Hmi.HmiTarget;
using Siemens.Engineering.Hmi.Tag;
using Siemens.Engineering.Hmi.Screen;
using Siemens.Engineering.Hmi.Cycle;
using Siemens.Engineering.Hmi.Communication;
using Siemens.Engineering.Hmi.Globalization;
using Siemens.Engineering.Hmi.TextGraphicList;
using Siemens.Engineering.Hmi.RuntimeScripting;
using System.Collections.Generic;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using System.IO;
using System.Security;
namespace ChangesInTheAPI
{
    internal class Program
    {
        private static void Main(string[] args)
        {
            // generate source for V14
            var blocks = new List<PlcBlock>() { block1 };
            var types = new List<PlcBlock>() { udt1 };
            var fileInfoBlock = new FileInfo(@"D:\Export\Block.scl");
            var fileInfoType = new FileInfo(@"D:\Export\Type.udt");
            PlcBlockSystemGroup blocksGroup = ...;
            blocksGroup.GenerateSourceFromBlocks(blocks, fileInfo);
            PlcTypeSystemGroup plcDataTypesGroup = ...;
            plcDataTypesGroup.GenerateSourceFromTypes(types, fileInfo);
            //generate source as of V14 SP1
            var blocks = new List<PlcBlock>(){ block1 };
            var types = new List<PlcBlock>() { udt1 };
            var fileInfoBlock = new FileInfo(@"D:\Export\Blocks.scl");
            var fileInfoType = new FileInfo(@"D:\Export\Type.udt");
            PlcExternalSourceSystemGroup externalSourceGroup = plc.ExternalSourceGroup;
            externalSourceGroup.GenerateSource(blocks, fileInfoBlock);
            externalSourceGroup.GenerateSource(types, fileInfoType);
        }
    }
}
```


7.6.4.3 Schemaerweiterung

Schemaerweiterung für Kommentare und Startwerte

Kommentare und Startwerte werden in dem neuen Element "Subelement" gespeichert, das sich auf das Array-Element mit dem Attribut "Path" bezieht.

Subelement enthält den Startwert und den Kommentar für das referenzierte Array-Element. Das Attribut "Path" an StartValue wird im neuen Schema entfernt.

Schemadefinition von "Subelement":

```
<xs:element name="Subelement" type="Subelement_T"/>
<xs:complexType name="Subelement_T">
  <xs:sequence>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="StartValue"/>
      <xs:element ref="Comment"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="Path" type="IndexPath_TP"/>
</xs:complexType>
```

Erweiterung des Mitgliedstyps:

```
<xs:complexType name="Member_T">
  <xs:sequence>
    <xs:element ref="AttributeList" minOccurs="0" maxOccurs="1"/>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="Member"/>
      <xs:element ref="Sections"/>
      <xs:element ref="StartValue"/>
      <xs:element ref="Comment"/>
      <xs:element ref="Subelement"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

Beispiele:

Speicherung von Kommentaren und Startwerten in einfachen Arrays:

```
<Member Name="Static_1" Datatype="Array[0..1] of Bool">
  <Comment>
    <MultiLanguageText Lang="de-DE">comment for array</MultiLanguageText>
  </Comment>
  <Subelement Path="0">
    <StartValue>true</StartValue>
    <Comment>
      <MultiLanguageText Lang="de-DE">comment for array element 0</MultiLanguageText>
    </Comment>
  </Subelement>
  <Subelement Path="1">
    <StartValue>true</StartValue>
    <Comment>
      <MultiLanguageText Lang="de-DE">comment for array element 1</MultiLanguageText>
    </Comment>
  </Subelement>
</Member>
```

Speicherung von Kommentaren und Startwerten in Arrays aus UDT:

```
<Member Name="Static_1" Datatype="Array[0..1] of &quot;User_data_type_1&quot;">
  <Comment>
    <MultiLanguageText Lang="de-DE">comment for array</MultiLanguageText>
  </Comment>
  <Subelement Path="0">
    <Comment>
      <MultiLanguageText Lang="de-DE">cmt array 0</MultiLanguageText>
    </Comment>
  </Subelement>
  <Sections>
    <Section Name="None">
      <Member Name="Element_1" Datatype="Bool">
        <Subelement Path="0">
          <StartValue>true</StartValue>
          <Comment>
            <MultiLanguageText Lang="de-DE">comment for element 0</MultiLanguageText>
          </Comment>
        </Subelement>
        <Subelement Path="1">
          <StartValue>true</StartValue>
          <Comment>
            <MultiLanguageText Lang="de-DE">comment for element 1</MultiLanguageText>
          </Comment>
        </Subelement>
      </Member>
      <Member Name="Element_2" Datatype="Struct">
        <Member Name="Element_1" Datatype="Int">
          <Subelement Path="0">
            <StartValue>11</StartValue>
            <Comment>
              <MultiLanguageText Lang="de-DE">comment for element 0</MultiLanguageText>
            </Comment>
          </Subelement>
        </Member>
      </Member>
    </Section>
  </Sections>
</Member>
```

Speicherung von Kommentaren und Startwerten in Arrays aus Struct:

```
<Member Name="Static_1" Datatype="Array[0..1] of Struct">
  <Member Name="Static_1" Datatype="Int">
    <Subelement Path="0">
      <StartValue>11</StartValue>
      <Comment>
        <MultiLanguageText Lang="de-DE">comment for int elem</MultiLanguageText>
      </Comment>
    </Subelement>
  </Member>
  <Member Name="Static_2" Datatype="Bool">
    <Subelement Path="1">
      <StartValue>true</StartValue>
      <Comment>
        <MultiLanguageText Lang="de-DE">comment for bool elem</MultiLanguageText>
      </Comment>
    </Subelement>
  </Member>
</Member>
```

7.6.4.4 Schemaänderungen

Knoten Access in SW.PlcBlocks.Access.xsd

Das Attribut Type des Knotens Access wurde zu den untergeordneten Knoten von Access verschoben bei

- AbsoluteOffset – erforderlich
- Address – optional

```
<StlStatement UId="22">
  <StlToken Text="L" />
  <Access Scope="Address">
    <Address Area="Local" Type="Word" BitOffset="80" />
  </Access>
</StlStatement>
```

Das Attribut Type von Constant wurde mit dem neuen Unterknoten ConstantType ersetzt.

```
<Access Scope="LocalConstant">
  <IntegerAttribute Name="NumBLs" Informative="true">5</IntegerAttribute>
  <Constant Name="LocalConstant_A">
    <ConstantType Informative="true">Int</ConstantType>
    <ConstantValue Informative="true">10</ConstantValue>
    <StringAttribute Name="Format" Informative="true">Dec_signed</StringAttribute>
  </Constant>
</Access>
```

Der Wert des Attributs Scope in Access wurde in TypedConstant umbenannt, wenn der ConstantValue einen typqualifizierten Wert enthält (z. B.: int#10).

Constant hat kein Attribut Type, wenn ConstantValue einen typqualifizierten Wert enthält (z. B.: int#10).

Lokale Variablen haben keinen Knoten Address, wenn Scope eine LocalVariable ist.

Wenn ein Access auf beliebiger Ebene in einem anderen Access verschachtelt ist, muss nur der äußere Access eine UId haben.

Knoten Address in SW.PlcBlocks.Access.xsd

Das Attribut BitOffset des Knotens Address ist optional geworden.

Die Deklarationen zum Exportieren des absoluten Zugriffs wurde wie in der folgenden Tabelle gezeigt geändert:

Bereich ab V14 SP1	Typ	Bausteinnummer	Bit-Offset	Beispiel
DB	Block_DB	obligatorisch	verboten	OPN %DB12
DB	unsortiert	vorhanden	obligatorisch	%DB100.DBX10.3
DB	unsortiert	nicht vorhanden	obligatorisch	%DB100.DBX10.3
L	unsortiert	verboten	obligatorisch	%LW10.0
I	unsortiert	verboten	obligatorisch	%I0.0
Q				%Q0.0
M				%M0.0
T	unsortiert	verboten	obligatorisch	%T0
C				%C1
Block_FC Block_FC	Block_FC Block_FB	obligatorisch	verboten	Aufruf %FB4, %DB5 Input_1 := %FC10 Aufruf %FB4, %DB5 Input_2 := %FB11
Peripherieeingang	unsortiert	verboten	obligatorisch	
Peripherieausgang	unsortiert	verboten	obligatorisch	

Knoten Area in SW.PlcBlocks.Access.xsd

Der Knoten Area hat eine vereinfachte Enum-Liste erhalten:

- LocalC und LocalIN wurden zu Local
- DBc, DBv, DBr wurden entfernt.

Knoten CallInfo in SW.PlcBlocks.Access.xsd

Das Attribut Name des Knotens CallInfo ist optional geworden.

Das Attribut BlockType des Knotens CallInfo ist erforderlich geworden.

+2.2.5 Anwenderbausteinaufrufe

Knoten Constant in SW.PlcBlocks.Access.xsd

Der Knoten Constant referenziert den Knoten CostantType mit minOccurs=0.

Der Knoten Constant referenziert den Knoten IntegerAttribute nicht mehr.

Knoten ConstantValue in SW.PlcBlocks.Access.xsd

Der Knoten ConstantValue erhält ein informatives Attribut.

Knoten Instruction in SW.PlcBlocks.Access.xsd

Der Knoten Instruction referenziert den Knoten Acces mit minOccurs=0.

Die Parameterattribute Section, Type und TemplateReference wurden bei Instruction gelöscht.

Knoten Parameter in SW.PlcBlocks.Access.xsd

Das Attribut SectionName des Knotens Parameter ist optional geworden.

Werte für Scope in SW.PlcBlocks.Access.xsd

Die Enum-Liste von Scope wurde erweitert um:

- TypedConstant
- AddressConstant
- LiteralConstant
- AlarmConstant
- Address
- Statusword
- Expression
- Call
- CallWithType

Knoten Statusword in SW.PlcBlocks.Access.xsd

Die Enum-Liste von Statusword wurde erweitert um:

- STW

Knoten ConstantType in SW.PlcBlocks.Access.xsd

Der neue Knoten ConstantType wurde mit dem optional verwendeten Attribut Informative eingeführt.

Knoten CallRef in SW.PlcBlocks.LADFBDD.xsd

Der Knoten CallRef wurde in Call umbenannt und hat keinen Unterknoten BooleanAttribute mehr.

Knoten InstructionRef in SW.PlcBlocks.LADFBDD.xsd

Der Knoten InstructionRef wurde durch den Knoten Part ersetzt.

Knoten Part in SW.PlcBlocks.LADFBFD.xsd

Der neue Knoten ConstantType wurde eingeführt und ersetzt den Knoten InstructionRef.

- Attribute: Name und Version
- Unterknoten: Der Unterknoten Instruction als neue Wahl neben dem vorhandenen Equation
- hat weder einen Unterknoten BooleanAttribute noch ein Attribut Gate.

Knoten Wire in SW.PlcBlocks.LADFBFD.xsd

Das Attribut Name des Knotens Wire wurde entfernt.

Knoten TemplateReference in SW.PlcBlocks.LADFBFD.xsd

Der Knoten TemplateReference wurde gelöscht.

Knoten StatementList in SW.PlcBlocks.STL.xsd

Enum-Liste von StatementList (STL_TE):

- L_STW wurde entfernt
- T_STW wurde entfernt

7.6.4.5 Verhaltensänderungen

Absoluter Zugriff

In V14 wurde der Import des absoluten Zugriffs für die meisten Kombinationen abgebrochen. Ab V14 SP1 funktioniert der Import des absoluten Zugriffs für die folgenden Bereiche:

- Eingang
- Ausgang
- Merker
- Zeit, sofern auf dem PLC unterstützt
- Zähler, sofern auf dem PLC unterstützt
- DB
- DI

Wenn ein symbolischer Zugriff und ein absoluter Zugriff gleichzeitig verwendet und nicht von der Prüfung des Schemas oder der Knotenart abgelehnt werden, ist der Import nur dann erfolgreich, wenn die Boxzugriffsinformationen erfolgreich aufgelöst werden. Wenn der symbolische Zugriff zu anderen Informationen führt als der absolute Zugriff, wird der Import abgelehnt.

```
<Access Scope="Address">  
  <Address Area="Memory" Type="Word" BitOffset="0" />  
</Access>
```

```

<CallInfo Name="Block_1" BlockType="FC">
  <Parameter Name="Input_1" Section="Input" Type="Int" />
  <Parameter Name="Input_1" Section="Input" Type="Int" />
  <!-- Import will be aborted because parameter name 'Input_1' is used more than once -->
  <Parameter Name="Output_1" Section="NotExisting" Type="Int" />
  <!-- Import will be aborted because the section 'NotExisting' can not be used. -->
  <Parameter Name="ENO" Section="Output" Type="Int" />
  <!-- Import will be aborted because the parameter name 'ENO' is restricted and can not be used. -->
</CallInfo>

```

Direkter DB-Zugriff

Ab V14 SP1 kann der indirekte DB-Zugriff nur dann importiert werden, wenn 'Offset', 'Typ' und 'Symbol' angegeben werden.

```

...
<Access Scope="LocalVariable" UId="21">
  <Symbol>
    <Component Name="Output_3" />
    <AbsoluteOffset BitOffset="16" Type="Word" />
  </Symbol>
</Access>
...

```

Symbolische und absolute Informationen für lokalen Zugriff

Beim Importieren von "symbolischem Zugriff" werden alle möglichen angegebenen "Informationen für absoluten Zugriff" überprüft, sofern sie nicht als "informativ" gekennzeichnet sind. Ab V14 SP1 wird der Import abgebrochen, wenn die absoluten Informationen nicht übereinstimmen.

Einschränkungen der Bausteinschnittstelle

In V14SP1 werden verschiedene Einschränkungen geprüft. Diese Einschränkungen sind Benutzern des Bausteinschnittstellen-Editors wohlbekannt. Immer wenn der Bausteinschnittstellen-Editor einen Parameter durch Ergänzen oder Erhöhen von '_1' umbenennt, wird der OPNS-Import abgebrochen.

Beispielsweise werden die folgenden Einschränkungen überprüft:

- Doppelte Parameternamen
- Falsche Abschnittsnamen, einschließlich 'Return-Abschnitt' bei FB-Bausteinen
- Eingeschränkte Wörter

Sortierung der Abschnitte beim Import

Wenn der aufgerufene Baustein zum Zeitpunkt des Imports nicht vorhanden ist, wird die Schnittstellendefinition auf Aufrufseite dazu verwendet, den aufgerufenen Anwenderbaustein anzuzeigen. In V14 SP1 werden die Abschnitte in der Reihenfolge sortiert, in der sie in der Bausteinschnittstelle des aufgerufenen Bausteins angezeigt werden würden, wenn dieser mit den gleichen Parametern vorhanden wäre.

Die Abschnittsreihenfolge der importierten Parameter lautet:

- Eingang
- Ausgang

Das folgende Beispiel in AWL xml

```
<StlStatement UId="21">
<StlToken Text="CALL" />
  <Access Scope="Call">
    <CallInfo Name="Block_2" BlockType="FC">
      <Parameter Name="Output_1" Section="Output" Type="Int">
        <Access Scope="GlobalVariable">
          <Symbol>
            <Component Name="Tag_3" />
          </Symbol>
        </Access>
      </Parameter>
      <Parameter Name="Input_1" Section="Input" Type="Int">
        <Access Scope="GlobalVariable">
          <Symbol>
            <Component Name="Tag_1" />
          </Symbol>
        </Access>
      </Parameter>
      <Parameter Name="Output_2" Section="Output" Type="Int">
        <Access Scope="GlobalVariable">
          <Symbol>
            <Component Name="Tag_4" />
          </Symbol>
        </Access>
      </Parameter>
      <Parameter Name="Input_2" Section="Input" Type="Int">
        <Access Scope="GlobalVariable">
          <Symbol>
            <Component Name="Tag_2" />
          </Symbol>
        </Access>
      </Parameter>
    </CallInfo>
  </Access>
</StlStatement>
```

führt zu:

CALL	"Block_2"
Input_1	:= "Tag_1"
Input_2	:= "Tag_2"
Output_1	:= "Tag_3"
Output_2	:= "Tag_4"

Eindeutige Aufrufernamen von Anwenderbausteinen

Im TIA Portal müssen Namen eindeutig sein. Das bedeutet beispielsweise, dass eine Variable nicht den gleichen Namen wie ein Baustein haben kann. Für den XML-Import der TIA Portal Openness API bedeutet dies für den Fall, dass die XML einen Anwenderbausteinaufruf enthält, bei dem der aufgerufene Baustein zum Zeitpunkt des Imports nicht vorhanden ist, der Name des aufgerufenen Bausteins verglichen mit allen vorhandenen Namen im Projekt eindeutig sein muss. Ist der Name des aufgerufenen Bausteins nicht eindeutig, wird der Import abgebrochen.

Im folgenden Beispiel wird der Import abgebrochen, weil der Name des aufgerufenen Bausteins "Tag_1" bereits für eine Variablentabelle verwendet wird.

```
...
<SW.Tags.PlcTag ID="1" CompositionName="Tags">
  <AttributeList>
    <DataTypeName>Int</DataTypeName>
    <LogicalAddress>%MW2</LogicalAddress>
    <Name>Tag_1</Name>
  </AttributeList>
</SW.Tags.PlcTag>
...
...
<StlStatement UId="21">
  <StlToken Text="CALL" />
  <Access Scope="Call">
    <CallInfo Name="Tag_1" BlockType="FC">
      <Parameter Name="Input_1" Section="Input" Type="Int">
        <Access Scope="GlobalVariable">
          <Symbol>
            <Component Name="Tag_1" />
          </Symbol>
        </Access>
      </Parameter>
    </CallInfo>
  </Access>
</StlStatement>
...
```

Im folgenden Beispiel wird der Import abgebrochen, weil zwei Parameter den gleichen Namen "Input1" haben.

```

<StlStatement Uid="22">
  <StlToken Text="CALL" />
  <Access Scope="Call">
    <CallInfo Name="Block_1" BlockType="FB">
      <Instance Scope="GlobalVariable">
        <Component Name="Block_1_DB" />
      </Instance>
      <Parameter Name="Input1" Section="Input" Type="Int">
        <Access Scope="GlobalVariable">
          <Symbol>
            <Component Name="Tag_9" />
          </Symbol>
        </Access>
      </Parameter>
      <Parameter Name="Input1" Section="Input" Type="Time">
        <Access Scope="TypedConstant">
          <Constant>
            <ConstantValue>T#1s</ConstantValue>
          </Constant>
        </Access>
      </Parameter>
    </CallInfo>
  </Access>
</StlStatement>

```

Bibliotheksbausteinaufrufe

Die importierte XML kann Aufrufe von Anwenderbausteinen enthalten. Diese Anwenderbausteine werden anhand des Namens identifiziert.

Anwenderbausteine können auch Bibliothekselemente aufrufen. Diese Bibliothekselemente können als 'Bibliotheksbausteinaufrufe' generiert werden. Da Bibliotheksbausteinen denselben Namensraum wie Anwenderbausteine verwenden, kann bei einem anhand des Namens durchgeführten Imports eines Anwenderbausteinaufrufs die Implementierung eines Anwenderbausteins aufgerufen werden.

Vor V14 SP1 hat der Importvorgang versucht, die Parameter zwischen dem Anwenderbausteinaufruf und dem Anweisungsbausteinaufruf zuzuordnen. Gelegentlich wurde der Import abgebrochen, gelegentlich wurden beim Import alle nicht übereinstimmenden Parameter gelöscht.

Ab V14 SP1 findet der Anwenderbausteinaufruf immer noch den Bibliotheksbaustein, doch der Aufruf wird nicht ungültig.

Nichtübereinstimmung des Bausteintyps

Wenn die XML einen Anwenderbausteinaufruf von 'Block_1' mit mehr Parametern als die entsprechende FC im Projekt enthält, definiert der Import ab V14 SP1 eine neue Schnittstelle für den aufgerufenen Baustein, die dem Anwenderbausteinaufruf aus der XML entspricht. Bei der nächsten Übersetzung des Programmbausteins wird versucht, den Aufruf zu aktualisieren.

Neue Funktionsumfänge für Konstanten

Ab V14SP1 wurden verschiedene neue Funktionsumfänge für Konstanten definiert. Der Import ist nur dann erfolgreich, wenn die Werte in der XML-Datei dem Funktionsumfang der Konstante entsprechen. Der Import wird möglicherweise abgebrochen, wenn nicht alle für eine Konstante angegebenen Informationen der vorhandenen Konstante entsprechen.

```
...
    <Access Scope="LiteralConstant">
        <Constant>
            <ConstantType>Int</ConstantType>
            <ConstantValue>16#0000_0001</ConstantValue>
        </Constant>
    </Access>
...
    <Access Scope="TypedConstant">
        <Constant>
            <ConstantValue>Int#10</ConstantValue>
        </Constant>
    </Access>
...
    <Access Scope="LiteralConstant">
        <Constant>
            <ConstantType>Int</ConstantType>
            <ConstantValue>10</ConstantValue>
        </Constant>
    </Access>
...
    <Access Scope="GlobalConstant">
        <Constant Name="Constant_1" />
    </Access>
...
    <Access Scope="LocalConstant">
        <Constant Name="Constant_1" />
    </Access>
...
    <Access Scope="AddressConstant">
        <Constant Name="Tag_1" />
    </Access>
...
    <Access Scope="AlarmConstant">
        <Constant>
            <ConstantType>C_Alarm</ConstantType>
            <ConstantValue>16#0000_0001</ConstantValue>
        </Constant>
    </Access>
...
```

Anmerkungen von Anweisungsversionen

Ab V14 SP1 können nur Anweisungsversionen importiert werden, die auf dem PLC einsetzbar sind, in den importiert werden soll. Ist in der XML keine Anweisungsversion angemerkt, wird die im PLC ausgewählte Version verwendet. In KOP und FUP gibt es für einige als Anweisungen dargestellte Elemente keine Versionierung. Diese Elemente können nur ohne Version importiert werden.

```
...
<Part Name="MIN" Version="1.0" UId="27" DisabledENO="false">
  <TemplateValue Name="card" Type="Cardinality">2</TemplateValue>
  <TemplateValue Name="value_type" Type="Type">Int</TemplateValue>
</Part>
<Part Name="MIN" UId="28" DisabledENO="false">
  <TemplateValue Name="card" Type="Cardinality">2</TemplateValue>
  <TemplateValue Name="value_type" Type="Type">Int</TemplateValue>
</Part>
...
```

Deaktivierter ENO

Die Funktion zum Deaktivieren des ENO dient auf S7-1200 und S7-1500 PLCs dazu, Laufzeit verbrauchende Berechnungen des ENO-Verbindungsstatus zu deaktivieren.

Ab V14 SP1 kann der Merker DisabledENO nur auf PLCs importiert werden, die diese Funktion unterstützen.

```
...
<Part Name="Add" UId="24" DisabledENO="false">
  <TemplateValue Name="Card" Type="Cardinality">2</TemplateValue>
  <TemplateValue Name="SrcType" Type="Type">Int</TemplateValue>
</Part>
...
```

Typvalidierung für absoluten Zugriff auf den L-Stack

Ab V14 SP1 wird der Import abgebrochen, wenn der Typ nicht verwendet oder zugeordnet werden kann.

Validierung von Index-Idents

Indezzugriff ist verwendbar, wenn die Funktion "Symbolischer Zugriff auf den Speicher" definiert ist, beispielsweise lokaler Zugriff, globaler Zugriff, indirekter Zugriff.

Wenn eine literale Konstante als Index verwendet wird, werden die Typen Ganzzahl mit und ohne Vorzeichen in Dint geändert. Ab V14 SP1 wird der Import abgebrochen, wenn ein Typ außerhalb des angegebenen Bereichs angegeben wird.

Sämtlicher Indezzugriff wird daraufhin geprüft, ob die Art des Zugriffs überhaupt als 'Indezzugriff' verwendet werden kann. Ab V14 SP1 wird der Import abgebrochen, wenn der definierte Indezzugriff nicht verwendet werden kann.

Sortierung der Elementreihenfolge

Ab V14 SP1 werden die Elemente in KOP und FUP in der Reihenfolge der Codeerzeugung sortiert, wo beim Export automatisch möglich. In einigen seltenen Fällen kann die exportierte XML nicht wieder importiert werden. In diesen Fällen muss entweder die XML angepasst werden oder die entsprechenden Netzwerke müssen gelöscht und neu programmiert werden. Doch die Reihenfolge von Leitern und Referenzen ist immer noch nicht zuverlässig.

Alarmkonstanten

Mit V14 SP1 wurden die Übersetzungsprüfungen auf gültige Alarmkonstanten erweitert. Es kann vorkommen, dass aufgrund einer in V14 mit defekten Alarmkonstanten importierten XML ein Projekt in V14 SP1 übersetzbar ist. Öffnen Sie in diesem Fall das relevante Netzwerk im KOP/FUP-Editor und löschen Sie den tatsächlichen Alarmoperanden. Der Editor erstellt automatisch eine neue gültige Alarmkonstante.

```
<FlgNet>
  <Parts>
    <Access Scope="AlarmConstant" UId="21">
      <Constant>
        <ConstantType>C_Alarm</ConstantType>
        <ConstantValue>16#0000_0002</ConstantValue>
      </Constant>
    </Access>
    <Call UId="22">
      <CallInfo Name="Block_1" BlockType="FB">
        <Instance UId="23" Scope="GlobalVariable">
          <Component Name="Block_1_DB" />
        </Instance>
        <Parameter Name="Input_1" Section="Input" Type="C_Alarm" />
      </CallInfo>
    </Call>
  </Parts>
  <Wires>
    <Wire UId="24">
      <Powerrail />
      <NameCon UId="22" Name="en" />
    </Wire>
    <Wire UId="25">
      <IdentCon UId="21" />
      <NameCon UId="22" Name="Input_1" />
    </Wire>
  </Wires>
</FlgNet>
```

Einschränkungen für Instanzen von Anwenderbausteinen und Anweisungen

In V14 war es möglich, FC-Anwenderbausteinaufrufe mit einer Instanz zu importieren und diese Aufrufe gelegentlich sogar zu übersetzen.

Ab V14 SP1 ist der Import von Instanzen nur möglich, wo Instanzen unterstützt werden. Bestehende Projekte mit Instanzen an FC-Anwenderbausteinaufrufen und Anweisungen lassen sich möglicherweise nicht mehr übersetzen. In diesem Fall muss der Aufruf gelöscht und neu programmiert werden. Jeder Versuch, eine Aufrufaktualisierung oder eine andere Art der automatischen Reparatur durchzuführen, schlägt fehl.

EnEno sichtbar

In V14 waren die EN- und ENO-Anschlüsse von 'InstructionRef' abhängig vom Merker ENENO verwendbar oder nicht.

Ab V14SP1 werden vom OPNS während des Imports basierend auf dem Element und der Verdrahtung entweder die EN- oder die ENO-Anschlüsse verwendet. Aufgrund dieser automatischen Erkennung ist eine unterschiedliche Verwendung von EN- und ENO-Anschlüssen festzustellen. Höchstwahrscheinlich zeigen nur die Boxen von IEC-Zeiten und IEC-Zählern gewisse Probleme.

UId-Zuweisung

Die Zuweisung von UIds zu Teilen, Zugriffen und Leitern ändert sich mit V14 SP1. Die UIds für Ausdrücke, CallInfo und Operanden müssen in einer Übersetzungseinheit eindeutig sein. Aus der TIA Portal-Perspektive sind die UIds in der XML Schlüssel ohne zusätzliche Bedeutung neben der Identifikation eines Elements.

Prüfen von Zeichenfolgen

Strengere Prüfungen bezüglich Anführungszeichen, Stellvertreterzeichen und Steuerzeichen werden für das Attribut Name durchgeführt beim Import von

- IntegerAttribute
- StringAttribute
- DateAttribute
- AutomaticTyped
- Component
- Invisible
- Label
- NameCon
- Negated
- TemplateValue
- CallInfo
- Instruction
- Parameter
- Part
- Step

Strengere Prüfungen bezüglich Stellvertreterzeichen und Steuerzeichen werden durchgeführt beim Import von

- Bausteintiteln und Netzwerken
- LineComment-Text
- Konstanten Zeichenfolgen (Typen String, WString, Char, Wchar)

Strengere Prüfungen bezüglich Stellvertreterzeichen und Steuerzeichen (Tab und neue Zeile zulässig) werden durchgeführt beim Import von

- Kommentaren und Netzwerken
- String-Attributen
- Knoten, die mehrsprachige Texte definieren wie Alarmtext, Comments
- Token-Texte

Nichtbeachtung von Groß-/Kleinschreibung bei Vorlagenoperationen und Parametern

Ab V14 SP1 wird die Nichtbeachtung von Vorlagenoperationen für Anweisungen und Aufruf- oder Anweisungsparameter importiert und automatisch korrigiert.

Der folgende Code wird importiert und der fehlerhafte Wert "Eq" wird in "EQ" korrigiert und der fehlerhafte Parameter "iN1" wird in "IN1" korrigiert:

```
<StlStatement UIId="22">
  <StlToken Text="CALL" />
  <Access Scope="Call">
    <Instruction Name="CompType">
      <TemplateValue Name="src_type" Type="Type">Variant</TemplateValue>
      <TemplateValue Name="relation" Type="Operation">Eq</TemplateValue>
      <Parameter Name="iN1">
        <Access Scope="GlobalVariable">
          <Symbol>
            <Component Name="Tag_12" />
          </Symbol>
        </Access>
      </Parameter>
    </Instruction>
  </Access>
</StlStatement>
```

In Aufrufen verwendete Multiinstanzen

Ab V14 SP1 wird der Import abgebrochen, wenn die in einem Aufruf verwendete Multiinstanz nicht vorhanden ist.

Der folgende Code zeigt ein XML-Beispiel, in dem die Multiinstanz im Schnittstellenabschnitt korrekt definiert ist:

```

<SW.Blocks.FB ID="0">
  <AttributeList>
    <Interface>
      <Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
        <Section Name="Input" />
        <Section Name="Output" />
        <Section Name="InOut" />
        <Section Name="Static">
          <!-- The next line must be present if multiinstance is used in code-->
          <Member Name="Static_1" Datatype=""Block_2"" />
        </Section>
      </Sections>
    </Interface>
  </AttributeList>
  ....
  <StlStatement UId="22">
    <StlToken Text="CALL" />
    <Access Scope="Call">
      <CallInfo BlockType="FB">
        <!-- Multiinstance usage-->
        <Instance Scope="LocalVariable">
          <Component Name="Static_1" />
        </Instance>
        <Parameter Name="Input_1" Section="Input" Type="Int">
          <Access Scope="GlobalVariable">
            <Symbol>
              <Component Name="Tag_9" />
            </Symbol>
          </Access>
        </Parameter>
      </CallInfo>
    </Access>
  </StlStatement>

```

Vorlagenkardinalitäten in AWL

In AWL haben die Vorlagenkardinalitäten für jede Anweisung einen festen Standardwert, der der einzig gültige Wert ist. Ab V14 SP1 wird der Import abgebrochen, wenn ein anderer Wert für die Kardinalität verwendet wird.

Indirekten Zugriff importieren

Ab V14 SP1 kann der indirekte Zugriff nur dort importiert werden, wo er übersetzt werden kann.

```

<StlStatement UId="22">
  <StlToken Text="L" />
  <Access Scope="Address">
    <Indirect Width="Word" Area="Memory">
      <Access Scope="LocalVariable">
        <Symbol>
          <Component Name="Temp_1" />
        </Symbol>
      </Access>
    </Indirect>
  </Access>
</StlStatement>

```


Statuswörter Importieren

Ab V14 SP1 kann das Statuswort nur bei Anweisungen importiert werden, bei denen es unterstützt wird.

- L - Unterstütztes Statuswort: STW
- T - Unterstütztes Statuswort: STW
- A - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- AN - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- O - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- ON - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- X - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- XN - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU

Hinweis

Die meisten Statuswörter sind nur bei S7-300 und S7-400 PLCs nützlich.

Leere Anweisungen

Der Import wird abgebrochen, wenn eine Anweisung keinen Knoten `<StlStatement/>` hat. Bei einer leeren Anweisung ergänzen Sie den Knoten `<StlToken Text="Empty_Line" />`.

Der Import wird abgebrochen, wenn eine leere Anweisung über Kommentare verfügt. Bei einer Anweisung nur mit Kommentaren verwenden Sie `<StlToken Text="COMMENT" />`.

```
<!-- Declaration of an empty statement -->
<StlStatement UId="23">
  <StlToken Text="EMPTY_LINE" />
</StlStatement>

<!-- Declaration of a statement with only comments-->
<StlStatement UId="22">
  <LineComment>
    <Text>Comment number 1</Text>
  </LineComment>
  <StlToken Text="COMMENT" />
</StlStatement>
```

7.6.4.6 Änderungen an Bausteinattributen

Änderungen an allgemeinen Attributen

AutoNumber hat einen neuen Standardwert (false) bei klassischen OBs.

HeaderVersion hat einen neuen Typ System.Version (anstelle von String).

IsKnowHowProtected wird auch auf benutzerdefinierte Datentypen angewendet.

ILibraryTypeInstance.ConnectedVersion, ILibraryTypeInstance.Dependencies, ILibraryTypeInstance.Dependents wurden aus der Tabelle der allgemeinen Attribute entfernt, weil sie weder in XML exportiert werden noch über API zugänglich sind.

MemoryLayout hat einen neuen Standardwert: Standard bei klassischen PLCs und Optimized bei Plus-PLCs.

Number wird auch auf benutzerdefinierte Datentypen angewendet, und es wird in XML dargestellt und ist auch über API zugänglich.

Änderungen an spezifischen Attributen

IsOnlyStoredInLoadMemory und IsWriteProtectedInAS sind jetzt für IDBofUDT schreibgeschützt, sofern der UDT zu einem Systembibliothekselement gehört.

OfSystemLibElement und OfSystemLibVersion gehören jetzt nicht mehr zu den allgemeinen, sondern zu den spezifischen Attributen.

OfSystemLibVersion hat einen neuen Typ System.Version (anstelle von String).

ParameterPassing bleibt Lesen/Schreiben bei FCs und FBs nur, wenn

- ProgrammingLanguage AWL ist und
- MemoryLayout Standard ist und
- die Schnittstelle leer ist

GraphVersion hat einen neuen Typ System.Version (anstelle von String).

Ein neues Attribut namens ExtensionBlockName wird für in Graph geschriebene FBs (ab Graph Version V4) eingeführt.

Ein neues Attribut namens InitialValuesAcquisition wird für in Graph geschriebene FBs (ab Graph Version V4) eingeführt.

Ein neues Attribut namens IsWriteProtected wird für Codebausteine eingeführt.

DownloadWithoutReinit ist jetzt schreibgeschützt und gilt auch für IDBofFBs.

Supervisions ist jetzt bei IDBofFBs schreibgeschützt.

Änderungen in Enums

Die Enum-Werte für ProgrammingLanguage wurden wie folgt geändert:

- Ein neuer Enum-Wert F_CALL wurde eingeführt.
- Ein neuer Enum-Wert Motion_DB wurde für das Technologieobjekt Motion eingeführt.
- GRAPH_SEQUENCE, GRAPH_ACTIONS, GRAPH_ADDINFOS wurden aus den Enums gelöscht. Sie wurden durch GRAPH ersetzt.

Die Enum-Werte für BlockType wurden wie folgt geändert:

- Die Werte OB, FC, DB, SFC wurden gelöscht, weil diese Enums lediglich beim Attribut InstanceOfType verwendet werden.

7.7 Die wichtigsten Änderungen in V14

7.7.1 Wesentliche Änderungen des Objektmodells

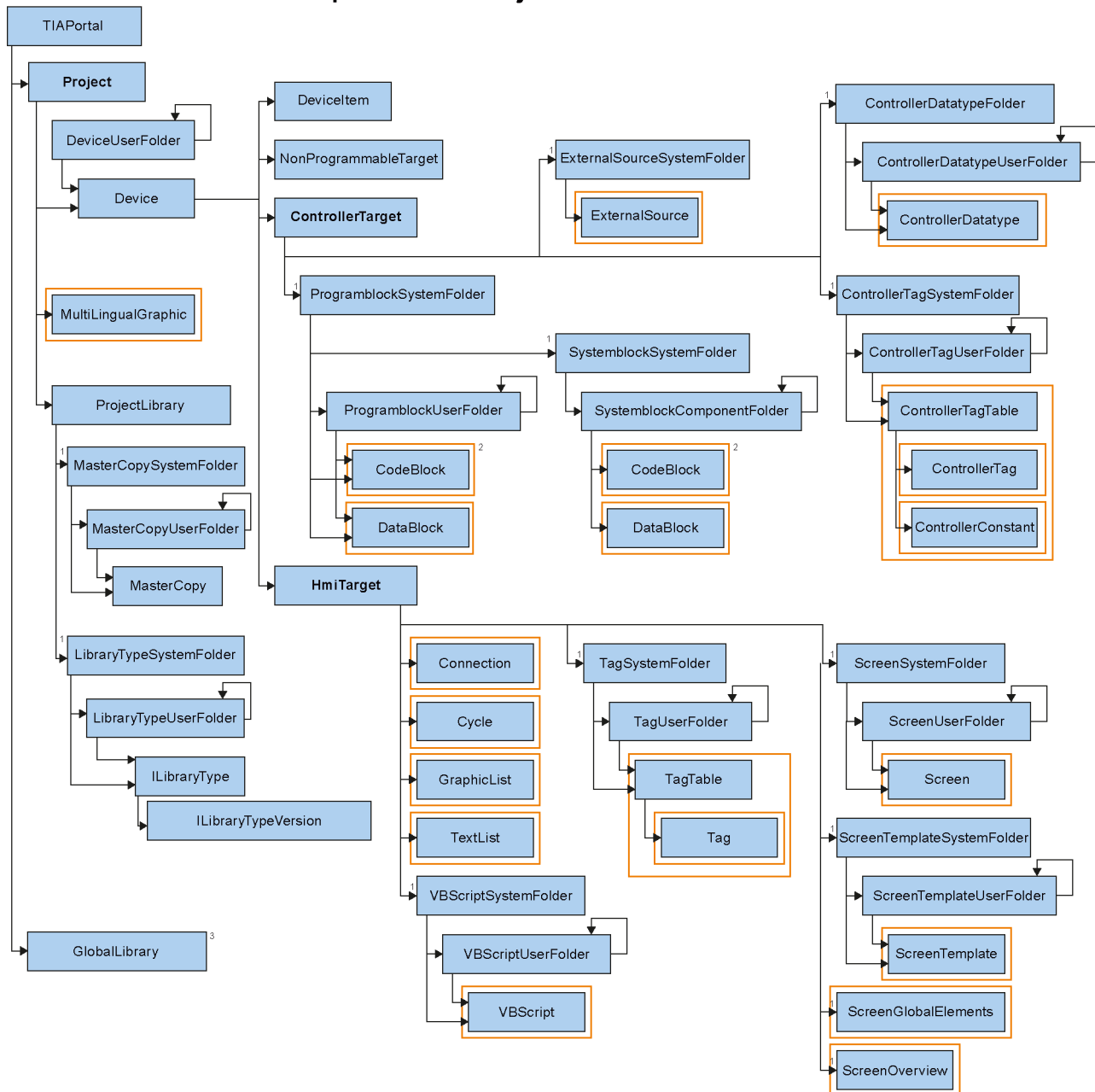
Objektmodell von TIA Portal Openness V13 SP1 und älter

Um Ihnen einen Vergleich zwischen dem alten und dem neuen Objektmodell von TIA Portal Openness zu ermöglichen, beschreibt das nachstehende Diagramm das Objektmodell von TIA Portal V13 SP1.

Hinweis

Das in dem Diagramm beschriebene Objektmodell ist veraltet. Sie finden Informationen über das Objektmodell von TIA Portal Openness V14 SP1 unter TIA Portal Openness-Objektmodell (Seite 55)

Openness object model V13 SP1



7.7.2 Vor dem Hochrüsten einer Anwendung auf TIA Portal Openness V14

Anwendung

Vor dem Hochrüsten einer Anwendung auf TIA Portal Openness V14 müssen die folgenden Einstellungen geändert werden:

1. Die Verweise auf die V14 API sind durch Ergänzen der folgenden TIA Portal Openness APIs anzupassen:
 - `Siemens.Engineering`
 - `Siemens.Engineering.Hmi`
2. Das .Net-Framework von Visual Studio auf Version 4.6.1 ändern
3. Aktualisieren Sie die AssemblyResolve-Methode durch Anpassen des neuen Installationspfads des TIA Portal.
 - Arbeiten Sie aus der Registrierungsdatei, passen Sie den neuen Schlüssel wie im folgenden Beispiel an:
`"HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\_InstalledSW\TIA AP14\TIA_Opns\..."`
 - Arbeiten Sie mit der Anwendungs-Konfigurationsdatei, passen Sie die Pfade an den neuen Installationspfad an.

7.7.3 Wesentliche Stringänderungen

Einleitung

An TIA Portal Openness V14 wurden die folgenden Änderungen vorgenommen, was möglicherweise Auswirkungen auf Ihre vorhandenen Anwendungen hat:

Änderung	Erforderliche Anpassung von Programmcode
Die Übersetzungsmethoden wurden geändert.	Ändern Sie die Übersetzungsmethoden wie im folgenden Beispiel: - TIA Portal Openness V13 SP1(veraltet): <code>controllerTarget.Compile(CompilerOptions. Software, BuildOptions.Rebuild);</code> - TIA Portal Openness V14: <code>plcSoftware.GetService<ICompilable>().Compile();</code>
Es wurden neue Namensräume hinzugefügt.	- Fügen Sie die folgenden Namensraumanweisungen hinzu: <code>Siemens.Engineering.SW.Blocks;</code> <code>Siemens.Engineering.SW.ExternalSources;</code> <code>Siemens.Engineering.SW.Tags;</code> <code>Siemens.Engineering.SW.Types;</code> - Entfernen Sie die Namensraumanweisung <code>using ControllerTarget =</code> <code>Siemens.Engineering.HW.ControllerTarget.</code> - Übersetzen Sie die Anwendung.
<code>ControllerTarget</code> wurde durch <code>PlcSoftware</code> ersetzt und in einigen Fällen hat sich Funktionalität geändert.	- Prüfen Sie die Codebeispiele in der Dokumentation, die zu Ihrer Anwendungsfunktionalität gehört. - Aktualisieren Sie den Programmcode Ihrer TIA Portal Openness-Anwendung nach dem folgenden Beispiel: - TIA Portal Openness V13 SP1(veraltet): <code>ControllerTarget controllerTarget = deviceItem as ControllerTarget</code> - TIA Portal Openness V14: <code>PlcSoftware plcSoftware = deviceItem.GetService<SoftwareContainer>().Software as PlcSoftware</code> - Übersetzen Sie die Anwendung.

Änderung	Erforderliche Anpassung von Programmcode
Es wurden Objekte ersetzt.	1. Suchen und ersetzen Sie die folgenden Objekte: <pre> DeviceUserFolderAggregation = DeviceUserGroupComposition DeviceFolders = DeviceGroups DeviceUserFolder = DeviceUserGroup ProgramblockSystemFolder = PlcBlockSystemGroup ProgramblockUserFolder = PlcBlockUserGroup IBlock = PlcBlock ControllerDatatypeSystemFolder = PlcTypeSystemGroup ControllerDatatypeUserFolder = PlcTypeUserGroup ControllerDatatype = PlcType ControllerTagSystemFolder = PlcTagTableSystemGroup ControllerTagUserFolder = PlcTagTableUserGroup ControllerTagTable = PlcTagTable ControllerTag = PlcTag ControllerConstant = PlcConstant ExternalSourceSystemFolder = PlcExternalSourceSystemGroup ExternalSource = PlcExternalSource IOOnline = OnlineProvider ILibraryType = LibraryType </pre> 2. Übersetzen Sie die Anwendung.
Aggregationen wurden durch Zusammensetzungen ersetzt.	1. Ersetzen Sie jede Aggregation in Ihrem Code durch Composition wie in den folgenden Beispielen: <pre> ProjectAggregation = ProjectComposition IDeviceAggregation = IDeviceComposition TagTableAggregation = TagTableComposition CycleAggregation = CycleComposition GraphicListAggregation = GraphicListComposition TextListAggregation = TextListComposition ConnectionAggregation = ConnectionComposition MultiLingualGraphicAggregation = MultiLingualGraphicComposition UpdateCheckResultMessageAggregation = UpdateCheckResultMessageComposition </pre> 2. Übersetzen Sie die Anwendung.
Ordner wurden in jeder Beziehung außer zu Bediengeräten durch Gruppen ersetzt.	1. Ersetzen Sie mit Ausnahme von Codeteilen, die Bediengeräte betreffen, jeden Folder in Ihrem Programmcode durch Group. 2. Übersetzen Sie die Anwendung.

Änderung	Erforderliche Anpassung von Programmcode
Die Methode <code>GetAttributeNames</code> wurde durch die Methode <code>GetAttributeInfos</code> ersetzt.	- Verwenden Sie <pre> IList<EngineeringAttributeInfo> IEngineeringObject.GetAttributeInfos(AttributeAccessMode attributeAccessMode); </pre> , um Attribute zu ermitteln. - Übersetzen Sie die Anwendung. Ausführlichere Informationen finden Sie unter Objektstruktur und -Attribute ermitteln (Seite 148).
Die Methode <code>Close</code> zum Schließen eines Objekts hat sich geändert.	- Ersetzen Sie <pre> project.Close(CloseMode.PromptIfModified); </pre> durch <code>project.Close();</code>. - Übersetzen Sie die Anwendung. Ausführlichere Informationen finden Sie unter Projekt schließen (Seite 162).
Gleichzeitiger Zugriff wurde durch exklusiven Zugriff und exklusive Transaktionen ersetzt.	- Ersetzen Sie den gleichzeitigen Zugriff durch exklusiven Zugriff und exklusive Transaktionen wie in den folgenden Beispielen: - TIA Portal Openness V13 SP1 (veraltet): <pre> tiaProject.StartTransaction("Reseting project to default"); ... tiaProject.CommitTransaction(); </pre> - TIA Portal Openness V14: <pre> //Use exclusive access to avoid user changes ExclusiveAccess exclusiveAccess = tiaPortal.ExclusiveAccess(); ... exclusiveAccess.Dispose(); //Use transaction to be able to rollbank changes: Transaction transaction = exclusiveAccess.Transaction(tiaProject, "Compiling device"); transaction.CommitOnDispose(); </pre> - Übersetzen Sie die Anwendung. Weitere Informationen hierzu siehe Ausschließlicher Zugriff (Seite 107) und Transaktionsbehandlung (Seite 118).

Änderung	Erforderliche Anpassung von Programmcode
Der Online-Zugriff auf die CPU wurde geändert	1. Ändern Sie den Online-Zugriff auf die CPU wie in den folgenden Beispielen: – TIA Portal Openness V13 SP1 (veraltet): <pre>((IOnline)controllerTarget).GoOffline();</pre> – TIA Portal Openness V14: <pre>((DeviceItem)plcSoftware.Parent.Parent).GetService<OnlineProvider>().GoOffline();</pre> 2. Übersetzen Sie die Anwendung.
Die Hardware-Konfiguration wurde geändert	1. Ändern Sie die Hardware-Konfiguration: <pre>Device.Elements = Device.Items</pre> 2. Entfernen Sie die folgenden Hardware-Attribute: – <code>Device.InternalDeviceItem</code> – <code>Device.SubType</code> 3. Übersetzen Sie die Anwendung.

Siehe auch

Umgang mit Exceptions (Seite 1229)

Verbindung zum TIA Portal aufbauen (Seite 84)

7.7.4 Importieren von Dateien, die mit TIA Portal Openness V13 SP1 und älter erzeugt wurden

Anwendung

Wenn Sie versuchen, Dateien zu importieren, die mit TIA Portal Openness V13 SP1 oder älter erzeugt wurden, wird eine Ausnahme wegen Inkompatibilität ausgelöst. Grund dafür sind Änderungen bei den HMI-Variablen und HMI-Bildschirmen. Die folgenden Tabellen zeigen die wichtigsten Attributänderungen. Ausführlichere Informationen finden Sie im Kapitel "Bilder erstellen Arbeiten mit Objekten und Objektgruppen > Arbeiten mit Objekten > Konfigurieren von Bereichen" in der Online-Hilfe des TIA Portal:

Änderungen von HMI-Variablen

Die folgende Tabelle zeigt die wichtigsten Änderungen der HMI-Variablenattribute:

Entfernte Attribute	Hinzugefügte Attribute
RangeMaximumType RangeMaximum RangeMinimumType RangeMinimum	LimitUpper2Type. LimitUpper2. LimitLower2Type. LimitLower2. LimitUpper1Type LimitUpper1 LimitLower1Type LimitLower1

Änderungen von HMI-Bildelementen

Die folgende Tabelle zeigt die wichtigsten Änderungen der Schieberegler-Attribute:

Entfernte Attribute	Hinzugefügte Attribute
	RangeLower1Color RangeLower1Enabled RangeLower2Color RangeLower2Enabled RangeNormalColor RangeNormalEnabled RangeUpper1Color RangeUpper1Enabled RangeUpper2Color RangeUpper2Enabled ScalePosition ShowLimitLines ShowLimitMarkers ShowLimitRanges

Die folgende Tabelle zeigt die wichtigsten Änderungen der Messbereichsattribute:

Entfernte Attribute	Hinzugefügte Attribute
DangerRangeColor	RangeLower1Color
DangerRangeStart	RangeLower1Enabled
DangerRangeVisible	RangeLower2Color
WarningRangeColor	RangeLower2Enabled
WarningRangeStart	RangeNormalColor
WarningRangeVisible	RangeNormalEnabled
	RangeUpper1Color
	RangeUpper1Enabled
	RangeUpper1Start
	RangeUpper2Color
	RangeUpper2Enabled
	RangeUpper2Start

Die folgende Tabelle zeigt die wichtigsten Änderungen der Balkenattribute:

Entfernte Attribute	Hinzugefügte Attribute
AlarmLowerLimitColor	RangeLower1Color
AlarmUpperLimitColor	RangeLower1Enabled
	RangeLower2Color
	RangeLower2Enabled
	RangeNormalColor
	RangeNormalEnabled
	RangeUpper1Color
	RangeUpper1Enabled
	RangeUpper2Color
	RangeUpper2Enabled

Index

A

Abfragen

- Attribut „Consistency“ eines Bausteins, 515
- Bausteinautor, 515
- Bausteinfamilie, 515
- Bausteinname, 515
- Bausteinnummer, 515
- Bausteintitel, 515
- Bausteintyp, 515
- Bausteinversion, 515
- Finden, 510
- Informationen aus einer PLC-Variablen-tabelle, 606
- Informationen des Anwenderdatentyps, 515
- Informationen des Bausteins, 515
- Ordner „Programmbausteine“, 509
- Systemordner für PLC-Variablen, 601
- Technologieobjekt, 550
- Zeitstempel eines Bausteins, 515

Allgemeine Einstellungen des TIA Portals, 137

Anwenderdatentyp

- Exportieren, 1395
- Importieren, 1434
- Informationen abfragen, 515
- Löschen, 534
- Quelle generieren, 529

Anwendungsbeispiel Public API, 79

Ausnahmen

- Beim Zugriff auf das TIA Portal über öffentliche APIs, 1229

B

Baustein

- Exportieren, 1395
- Gruppe erzeugen, 519
- Gruppe löschen, 520
- Importieren, 1394
- Informationen abfragen, 515
- Löschen, 518
- Quelle generieren, 529

Bausteineditor

- Starten, 536

Bearbeitungssituation

- Ihre Openness-Anwendung und das TIA Portal befinden sich auf demselben Computer, 49

Beenden der Verbindung zum TIA Portal, 101

Beispielprogramm, 73

Besondere Überlegungen für HMI-Variablen des Datentyps "UDT", 1278

Bibliothek

- Auf Ordner zugreifen, 224
- Funktionen, 205
- Typversionen von Instanzen ermitteln, 246

C

CFC-Pläne

- Exportieren, 1328, 1329, 1332, 1333, 1335, 1336
- Importieren, 1331

D

Datentypen

- Technologieobjekt, 549

E

Editor „Geräte & Netze“

- Öffnen, 276

Enumerieren

- Alle Variablen einer Variablen-tabelle, 670
- Bausteine, 513
- Benutzerdefinierte Bausteinordner, 512, 555
- Benutzerdefinierte Ordner für PLC-Variablen, 602
- Geräte, 353, 356
- Geräteelemente, 375
- Mehrsprachige Texte, 146, 151
- Parameter eines Technologieobjekts, 558
- PLC-Variablen, 609
- PLC-Variablen-tabellen, 605
- Systemunterordner, 510
- Technologieobjekt, 555

Erstellen

- Benutzerdefinierte Bilderordner erzeugen, 664
- Benutzerdefinierte Ordner für HMI-Variablen, 669
- Benutzerdefinierte Unterordner für Skripte, 672
- Benutzerdefinierter Ordner für PLC-Variablen-tabellen, 603
- Gruppe für Baustein, 519
- Messtaster, 569
- Nocken, 569
- Nockenspur, 569
- Technologieobjekt, 551, 552

Export/Import

- Verwendung, 53

Exportdatei

- Grundstruktur, 1265, 1480, 1483

- Inhalt, 1254

- Struktur der XML-Datei, 1265, 1483

Exportierbare Bildobjekte, 1290

Exportieren

- Anwenderdatentyp, 1395

- Baustein, 1395

- CFC-Pläne, 1328, 1329, 1332, 1333, 1335, 1336

- Einzelne Variable oder Konstante aus einer PLC-Variablen-tabelle, 1475

- Technologieobjekte, 1453

F

Finden

- Messtaster, 569

- Nocken, 569

- Nockenspur, 569

- Parameter eines Technologieobjekts, 559

- Technologieobjekt, 557

Funktionen, 73

- Alle Bilder löschen, 666

- Allgemein, 84, 97, 101

- Allgemeine Einstellungen des TIA Portals, 137

- Anwendungsbeispiel Public API, 79

- Attribut „Consistency“ eines Bausteins abfragen, 515

- Bausteinautor abfragen, 515

- Bausteine enumerieren, 513

- Bausteinfamilie abfragen, 515

- Bausteinname abfragen, 515

- Bausteinnummer abfragen, 515

- Bausteintitel abfragen, 515

- Bausteintyp abfragen, 515

- Bausteinversion abfragen, 515

- Begrenzung auf Projekte von TIA Portal V13, 131

- Benutzerdefinierte Bausteinordner enumerieren, 512, 555

- Benutzerdefinierte Bilderordner erzeugen, 664

- Benutzerdefinierte Ordner für HMI-Variablen erzeugen, 669

- Benutzerdefinierte Ordner für PLC-Variablen enumerieren, 602

- Benutzerdefinierte Unterordner für Skripte erzeugen, 672

- Benutzerdefinierten Ordner für PLC-Variablen-tabellen erzeugen, 603

- Benutzerdefinierten Ordner für PLC-Variablen-tabellen löschen, 604

- Bild löschen, 664

- Bildvorlage löschen, 665

- Einlesen der Uhrzeit der letzten Änderungen in eine PLC-Variablen-tabelle, 607

- Geräte enumerieren, 353, 356

- Geräteelemente enumerieren, 375

- Grafikliste löschen, 668

- Grafiksammlungen löschen, 156

- HMI, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673

- Informationen aus einer PLC-Variablen-tabelle abfragen, 606

- Mehrsprachige Texte enumerieren, 146, 151

- Ordner „Programm-bausteine“ abfragen, 509

- PLC, 509, 510, 512, 513, 515, 555, 603, 604, 607, 608, 609, 611, 1474, 1475, 1477

- PLC-Konstanten, 611

- PLC-Target und HMI-Target abfragen, 277

- PLC-Variablen enumerieren, 609

- PLC-Variablen-tabelle importieren, 1474

- PLC-Variablen-tabelle löschen, 608

- PLC-Variablen-tabellen in Ordnern enumerieren, 605

- Projekt öffnen, 131

- Projekt schließen, 162

- Projekt speichern, 160

- Projekte, 131, 137, 146, 151, 156, 160, 162, 277, 353, 356, 375, 601, 602, 605, 606, 609

- Systemordner ermitteln, 510

- Systemordner für PLC-Variablen abfragen, 601

- Systemunterordner enumerieren, 510

- Textliste löschen, 667

- Variable aus einer PLC-Variablen-tabelle löschen, 609

- Variable aus einer Variablen-tabelle löschen, 671

- Variable in eine PLC-Variablen-tabelle importieren, 1477

- Variable oder Konstante aus einer PLC-Variablen-tabelle exportieren, 1475

- Variablen einer HMI-Variablen-tabelle enumerieren, 670

- Variablen-tabelle löschen, 671

- VB-Skript aus einem Ordner löschen, 673

- Verbindung löschen, 669

- Zeitstempel eines Bausteins abfragen, 515

- Zyklus löschen, 667

G

Generieren

- Quelle aus Anwenderdatentyp, 529

- Quelle aus Baustein, 529

Geräte enumerieren, 353, 356
 Geräteelemente enumerieren, 375
 Globale Bibliothek
 Zugreifen, 206, 211
 Zugriff auf Spracheinstellungen, 209
 Grundstruktur einer AML-Exportdatei, 1480
 Grundstruktur einer Exportdatei, 1265, 1483

H

Hardware
 Übersetzen, 157
 Hierarchie von Hardware-Objekten des
 Objektmodells, 67
 HMI-Variablen des Datentyps "UDT", 1278

I

Import/Export
 XML-Datei bearbeiten, 1253
 Importieren
 Anwenderdatentyp, 1434
 Baustein, 1394
 CFC-Pläne, 1331
 Einzelne Variable in eine PLC-
 Variablentabelle, 1477
 PLC-Variablentabellen, 1474
 Technologieobjekte, 1455
 Importieren/Exportieren
 Alle Bildvorlagen exportieren, 1303
 Alle Grafiken eines Projekts exportieren, 1258
 Anwendungsbereich, 1252
 Auch Standardwerte exportieren, 1254
 Ausgewählte Variable exportieren, 1275
 Bausteine mit Knowhow-Schutz
 exportieren, 1352
 Bausteine ohne Knowhow-Schutz
 exportieren, 1395
 Besondere Überlegungen für integrierte HMI-
 Variablen, 1277
 Bild aus einem Bilderordner exportieren, 1297
 Bild mit einer Bildbausteininstanz
 exportieren, 1320
 Bild mit einer Bildbausteininstanz
 importieren, 1322
 Bilder eines HMI-Geräts exportieren, 1294
 Bilder in ein HMI-Gerät importieren, 1299
 Bildvorlagen exportieren, 1307
 Bildvorlagen importieren, 1310
 CFC-Pläne, 1328, 1329, 1331, 1332, 1333, 1335,
 1336

Datenstruktur, 1265, 1483
 Eine HMI-Variable in eine Variablentabelle
 importieren, 1276
 Einschränkungen, 1251
 Erweiterte XML-Formate für den Export/Import von
 Textlisten, 1284
 Exporte auf geänderte Werte beschränken, 1254
 Exporteinstellungen, 1253
 Exportformat, 1251
 Exportierbare Bildobjekte, 1290
 Exportierbare Objekte, 1249
 Exportieren eines Slide-in-Bilds, 1317
 Exportieren mehrsprachiger Kommentare, 1626,
 1635, 1643, 1645
 Exportieren von Pop-up-Bildern, 1314
 Exportumfang, 1253
 Grafiken, 1257
 Grafiken in ein Projekt importieren, 1259
 Grafikliste importieren, 1287
 Grafiklisten exportieren, 1287
 Grundlagen, 1249
 HMI, 1269, 1270, 1271, 1274, 1275, 1276, 1277,
 1279, 1280, 1282, 1284, 1287, 1289, 1290, 1294,
 1297, 1299, 1302, 1303, 1307, 1310, 1314, 1316,
 1317, 1318, 1320, 1322, 1473
 HMI-Variablentabellen exportieren, 1271
 Importierbare Objekte, 1249
 Importieren eines Slide-in-Bilds, 1318
 Importieren mehrsprachiger Kommentare, 1626,
 1635, 1643, 1645
 Importieren von Pop-up-Bildern, 1316
 Importverhalten mit Programmcodes
 einstellen, 1256
 Konfigurationsdaten exportieren, 1253
 Konfigurationsdaten importieren, 1255
 Nur geänderte Werte exportieren, 1254
 Objekte von AML, 1480
 Permanentfenster exportieren, 1302
 Permanentfenster importieren, 1302
 PLC, 1352, 1391, 1395
 PLC-Variablentabelle exportieren, 1473
 Projektdaten, 1258, 1259
 Round-Trip-Geräte und -Module, 1580
 Stabile AML-GUIDs, 1580
 Systembausteine exportieren, 1391
 Technologieobjekte, 1453, 1455
 Textlisten exportieren, 1282
 Textlisten importieren, 1284
 Variable aus einer Variablentabelle
 exportieren, 1275
 Variablen exportieren, 1659
 Variablen importieren, 1659

- Variablentabelle in einen Variablenordner importieren, 1274
- VB-Skripte exportieren, 1279
- VB-Skripte importieren, 1280
- Verbindungen exportieren, 1288
- Verbindungen importieren, 1289
- Vorgehensweise zum Importieren, 1256
- Zyklen exportieren, 1269
- Zyklen importieren, 1270

Installation

- Authentifizierung für Zugriff prüfen, 41
- Benutzer der Benutzergruppe hinzufügen, 41
- Standardschritte für den Zugriff auf das TIA Portal, 47
- TIA Openness V13 Add-on-Paket, 40

- Installation des Add-on-Pakets, 40

Instanzen

- Typversion ermitteln, 246

- Integrierte HMI-Variablen, 1277

K

Konfiguration

- Ihre Openness-Anwendung und das TIA Portal befinden sich auf unterschiedlichen Computern, 49

Kopieren

- Inhalt einer Masterkopie in Projektordner, 243
- Masterkopie, 246

L

Lesen

- Parameter eines Technologieobjekts, 560
- Uhrzeit der letzten Änderungen an einer PLC-Variablentabelle, 607

Löschen

- Alle Bilder, 666
- Anwenderdatentyp, 534
- Baustein, 518
- Benutzerdefinierter Ordner für PLC-Variablentabellen, 604
- Bild, 664
- Bildvorlage, 665
- Einzelne Variable in einer PLC-Variablentabelle, 609
- Einzelne Variablen einer Variablentabelle, 671
- Grafikliste, 668
- Grafiksammlung, 156
- Gruppe für Baustein, 520
- PLC-Konstanten, 611

- PLC-Variablentabelle aus einem Ordner löschen, 608
- Programmbaustein, 518
- Technologieobjekt, 552, 553
- Textliste, 667
- Variablentabelle, 671
- VB-Skript aus einem Ordner, 673
- Verbindung, 669
- Zyklus, 667

M

Masterkopie

- Inhalt in Projektordner kopieren, 243
- Kopieren, 246

Masterkopien

- Löschen, 267

- Mehrsprachige Texte enumerieren, 146, 151

O

Objekte

- Exportierbare Objekte, 1249
- Importierbare Objekte, 1249

- Objektmodell, 55

Öffnen

- Editor „Geräte & Netze“, 276

- Öffnen eines Projekts, 131

Ordner

- Löschen, 267

P

Parameter

- Easy Motion Control, 600
- S7-1500 Motion Control, 571, 573, 575, 576, 577, 578
- Zählung, 599

Parameter eines Technologieobjekts

- Enumerieren, 558
- Finden, 559
- Lesen, 560
- Schreiben, 561

PLC

- Online-Verbindung herstellen, 479
- Online-Verbindung trennen, 479
- Status ermitteln, 450
- Vergleich mit tatsächlichem Status, 434
- Vergleichen, 434

Programmbaustein

- Löschen, 518

Programmgesteuertes Quittieren von Systemereignissen, 97

Programmierübersicht, 73

Projekt

Gerätetyp abfragen, 277

HMI-Targets abfragen, 277

Öffnen, 131

PLC-Targets abfragen, 277

Schließen, 162

Speichern, 160

Projekt speichern, 160

Projektbibliothek

Auf Masterkopien zugreifen, 239

Zugreifen, 206, 211

S

Schreiben

Parameter eines Technologieobjekts, 561

Siemens.Engineering, 76

Siemens.Engineering.Hmi, 76

Siemens.Engineering.Hmi.Communication, 76

Siemens.Engineering.Hmi.Cycle, 76

Siemens.Engineering.Hmi.Globalization, 76

Siemens.Engineering.Hmi.RuntimeScripting, 76

Siemens.Engineering.Hmi.Screen, 76

Siemens.Engineering.Hmi.Tag, 76

Siemens.Engineering.Hmi.TextGraphicList, 76

Siemens.Engineering.HW, 76

Siemens.Engineering.SW, 76

Software

Übersetzen, 157

Starten

Bausteineditor, 536

Variableneditor, 601

Status (PLC)

Ermitteln, 450

Struktur der Exportdaten, 1265, 1480, 1483

T

Technologieobjekt, 547, 1448

Abfragen, 550

Datentypen, 549

Enumerieren, 555

Erstellen, 551, 552

Finden, 557

Löschen, 552, 553

Übersetzen, 554

Technologieobjekte

Exportieren, 1453

Importieren, 1455

Technologieobjektgruppe

Übersetzen, 554

TIA Portal Openness, 51

Benutzer der Benutzergruppe hinzufügen, 41

Einleitung, 50

Export/Import, 53

Funktionen, 73

Funktionsumfang, 50

Grundbegriffe von Aggregationen, 115

Grundlegende Konzepte bei der Handhabung von Ausnahmen, 1229

Grundlegende Konzepte der Überprüfung auf Objektgleichheit, 116

Grundlegende Konzepte von Zuordnungen, 114

Konfiguration, 48

Notwendige Kenntnisse des Benutzers, 39

Programmierübersicht, 73

Public API, 73

Standardschritte für den Zugriff auf das TIA Portal, 47

Typische Aufgaben, 52

Voraussetzungen, 39

Zugriff, 52

Zugriffsrechte, 41

Typen

Löschen, 267

U

Übersetzen

Hardware, 157

Software, 157

Technologieobjekt, 554

Technologieobjektgruppe, 554

V

Variableneditor

Starten, 601

Verbindung zum TIA Portal

Einrichtung, 84

Schließen, 101

Verbindung zum TIA Portal herstellen, 84

Verschalten

Analogantriebe mit der Hardware-Adresse, 565

Analogantriebe mit einem Datenbaustein, 567

Antriebe, 579

Geber, 587

- Geber für Analogantriebe mit der Hardware-Adresse, 566
- Geber für PROFIdrives mit der Hardware-Adresse, 564
- Geber mit einem Datenbaustein, 568
- Gleichlaufachse mit Leitwerten, 592
- Messtaster, 591
- Nocken, 589
- Nockenspur, 589
- PROFIdrives mit der Hardware-Adresse, 563
- PROFIdrives mit einem Datenbaustein, 567
- PTO-Ausgang, 562
- Telegramm 750, 584

X

- XML-Datei
 - Bearbeiten, 1253
 - Export, 1254

Z

- Zugreifen
 - Masterkopie in einer Projektbibliothek, 239