

SIEMENS

SIMATIC

Automatisation de projets avec scripts

Manuel système

<u>Consignes de sécurité</u>	1
<u>Lisezmoi TIA Portal Openness</u>	2
<u>Nouveautés de TIA Portal Openness</u>	3
<u>notions de base</u>	4
<u>Introduction</u>	5
<u>Configurations</u>	6
<u>TIA Portal Openness API</u>	7
<u>Exportation/importation</u>	8
<u>Principales modifications</u>	9

Impression de l'aide en ligne

12/2017

Printout of the online help

Mentions légales

Signalétique d'avertissement

Ce manuel donne des consignes que vous devez respecter pour votre propre sécurité et pour éviter des dommages matériels. Les avertissements servant à votre sécurité personnelle sont accompagnés d'un triangle de danger, les avertissements concernant uniquement des dommages matériels sont dépourvus de ce triangle. Les avertissements sont représentés ci-après par ordre décroissant de niveau de risque.

 DANGER

signifie que la non-application des mesures de sécurité appropriées entraîne la mort ou des blessures graves.
--

 ATTENTION
--

signifie que la non-application des mesures de sécurité appropriées peut entraîner la mort ou des blessures graves.
--

 PRUDENCE

signifie que la non-application des mesures de sécurité appropriées peut entraîner des blessures légères.

IMPORTANT

signifie que la non-application des mesures de sécurité appropriées peut entraîner un dommage matériel.

En présence de plusieurs niveaux de risque, c'est toujours l'avertissement correspondant au niveau le plus élevé qui est reproduit. Si un avertissement avec triangle de danger prévient des risques de dommages corporels, le même avertissement peut aussi contenir un avis de mise en garde contre des dommages matériels.

Personnes qualifiées

L'appareil/le système décrit dans cette documentation ne doit être manipulé que par du **personnel qualifié** pour chaque tâche spécifique. La documentation relative à cette tâche doit être observée, en particulier les consignes de sécurité et avertissements. Les personnes qualifiées sont, en raison de leur formation et de leur expérience, en mesure de reconnaître les risques liés au maniement de ce produit / système et de les éviter.

Utilisation des produits Siemens conforme à leur destination

Tenez compte des points suivants:

 ATTENTION
--

Les produits Siemens ne doivent être utilisés que pour les cas d'application prévus dans le catalogue et dans la documentation technique correspondante. S'ils sont utilisés en liaison avec des produits et composants d'autres marques, ceux-ci doivent être recommandés ou agréés par Siemens. Le fonctionnement correct et sûr des produits suppose un transport, un entreposage, une mise en place, un montage, une mise en service, une utilisation et une maintenance dans les règles de l'art. Il faut respecter les conditions d'environnement admissibles ainsi que les indications dans les documentations afférentes.

Marques de fabrique

Toutes les désignations repérées par © sont des marques déposées de Siemens AG. Les autres désignations dans ce document peuvent être des marques dont l'utilisation par des tiers à leurs propres fins peut enfreindre les droits de leurs propriétaires respectifs.

Exclusion de responsabilité

Nous avons vérifié la conformité du contenu du présent document avec le matériel et le logiciel qui y sont décrits. Ne pouvant toutefois exclure toute divergence, nous ne pouvons pas nous porter garants de la conformité intégrale. Si l'usage de ce manuel devait révéler des erreurs, nous en tiendrons compte et apporterons les corrections nécessaires dès la prochaine édition.

Sommaire

1	Consignes de sécurité	11
2	Lisezmoi TIA Portal Openness	13
2.1	Lisezmoi.....	13
2.2	Modifications majeures dans TIA Portal Openness V15.....	16
2.3	Notification des modifications majeures dans les prochaines versions.....	18
2.4	Remarques sur l'écriture d'un code stable à long terme.....	19
3	Nouveautés de TIA Portal Openness	21
4	notions de base	23
4.1	Conditions requises pour TIA Portal Openness.....	23
4.2	Installation.....	25
4.2.1	Installation de TIA Openness.....	25
4.2.2	Ajouter un utilisateur au groupe d'utilisateurs "Siemens TIA Openness".....	26
4.2.3	Accéder au portail TIA.....	32
4.3	Tâches d'Openness.....	33
4.3.1	Possibilités d'utilisation.....	33
4.3.2	Exportation/importation.....	34
4.4	Liste d'objets.....	35
4.5	Bibliothèques standard.....	40
4.6	Remarques sur la performance de TIA Portal Openness.....	41
5	Introduction	43
6	Configurations	45
7	TIA Portal Openness API	49
7.1	Introduction.....	49
7.2	Etapes de programmation.....	50
7.3	Modèle d'objet TIA Portal Openness.....	51
7.4	Blocs et types de modèle d'objet TIA Portal Openness.....	56
7.5	Hiérarchie des objets matériels du modèle d'objet.....	64
7.6	Informations sur les versions de TIA Portal Openness installées.....	66
7.7	Exemple de programme.....	67
7.8	Utilisation des exemples de code.....	72
7.9	Fonctions générales.....	74
7.9.1	Prise en charge d'IntelliSense par TIA Portal Openness.....	74
7.9.2	Etablissement d'une connexion au portail TIA.....	74
7.9.3	Pare-feu TIA Portal Openness.....	79

7.9.4	Gestionnaire d'événements.....	80
7.9.5	Confirmer les boîtes de dialogue comportant des alarmes système par commande du programme.....	82
7.9.6	Mettre fin à la connexion au portail TIA.....	84
7.9.7	Interfaces de diagnostic dans TIA Portal.....	85
7.9.8	Exclusive access.....	90
7.9.9	Traitement des transactions.....	92
7.9.10	Créer un objet DirectoryInfo/FileInfo.....	95
7.9.11	Prise en charge de l'autodescription pour attributs, navigateurs, actions et services.....	96
7.10	Fonctions des projets et données de projet.....	99
7.10.1	Ouvrir un projet.....	99
7.10.2	Créer un projet.....	102
7.10.3	Accéder à des paramètres généraux de TIA Portal.....	104
7.10.4	Accéder à des langues.....	107
7.10.5	Déterminer la structure et les attributs de l'objet.....	109
7.10.6	Accéder au logiciel cible	111
7.10.7	Accéder à des textes multilingues et les énumérer.....	112
7.10.8	Compiler le projet.....	113
7.10.9	Lire des attributs liés au projet.....	116
7.10.10	Suppression d'un graphique du projet.....	118
7.10.11	Enregistrer le projet.....	119
7.10.12	Fermer un projet.....	120
7.11	Fonctions pour connexions.....	121
7.11.1	Attributs configurables d'une liaison port à port.....	121
7.12	Fonctions pour bibliothèques.....	125
7.12.1	Fonctions pour objets et instances.....	125
7.12.2	Accès aux bibliothèques globales.....	126
7.12.3	Ouvrir des bibliothèques.....	128
7.12.4	Énumérer des bibliothèques ouvertes.....	130
7.12.5	Enregistrer et fermer les bibliothèques.....	130
7.12.6	Créer des bibliothèques globales.....	131
7.12.7	Accéder aux dossiers dans une bibliothèque.....	132
7.12.8	Accéder aux types.....	135
7.12.9	Accéder aux types de versions.....	138
7.12.10	Accéder aux instances.....	142
7.12.11	Accéder à des modèles de copie.....	144
7.12.12	Créer la copie maîtresse d'un projet dans la bibliothèque.....	147
7.12.13	Créer un objet à partir d'une copie maîtresse.....	148
7.12.14	Copier des copies maîtresse.....	150
7.12.15	Déterminer les instances de type obsolètes.....	151
7.12.16	Actualiser un projet.....	154
7.12.17	Actualiser une bibliothèque.....	156
7.12.18	Supprimer les contenus de bibliothèque.....	157
7.13	Fonctions pour l'appel d'appareils, de réseaux et de liaisons.....	160
7.13.1	Ouvrir l'éditeur "Appareils & réseaux".....	160
7.13.2	Interroger PLC Target et HMI Target.....	161
7.13.3	Accéder à des attributs d'un objet adresse.....	162
7.13.4	Appeler une voie de module.....	165
7.13.5	Utilisation d'associations.....	167
7.13.6	Utilisation de compositions.....	167

7.13.7	Vérifier l'égalité des objets.....	168
7.13.8	Opérations de lecture pour attributs.....	169
7.14	Fonctions dans les réseaux.....	171
7.14.1	Créer un sous-réseau.....	171
7.14.2	Accéder aux sous-réseaux.....	172
7.14.3	Accéder à des sous-réseaux internes.....	173
7.14.4	Appeler l'identifiant de type de sous-réseaux.....	174
7.14.5	Appeler les attributs d'un sous-réseau.....	175
7.14.6	Supprimer un sous-réseau global.....	181
7.14.7	Énumérer tous les abonnés d'un sous-réseau.....	181
7.14.8	Énumérer les réseaux IO d'un sous-réseau.....	182
7.14.9	Accéder aux abonnés.....	183
7.14.10	Appeler les attributs d'un abonné.....	184
7.14.11	Relier l'abonné à un sous-réseau.....	188
7.14.12	Déconnecter un abonné d'un sous-réseau.....	188
7.14.13	Créer un réseau IO.....	189
7.14.14	Appeler les attributs d'un réseau IO.....	190
7.14.15	Relier IO-Connector à un réseau IO.....	190
7.14.16	Appeler le système maître ou le réseau IO d'une interface.....	191
7.14.17	Appeler un contrôleur IO.....	192
7.14.18	Appeler un connecteur IO.....	193
7.14.19	Déconnecter IO-Connector d'un réseau IO ou d'un réseau maître DP.....	193
7.14.20	Appeler un réseau maître DP.....	194
7.14.21	Appeler les attributs d'un réseau PROFINET IO.....	195
7.14.22	Supprimer un réseau maître DP.....	196
7.14.23	Supprimer un réseau Profinet IO.....	197
7.14.24	Créer un réseau maître DP.....	197
7.14.25	Appeler les informations de lien des ports de l'élément d'appareil du port.....	198
7.14.26	Attributs de la connexion de port.....	199
7.14.27	Appeler les attributs d'un port.....	202
7.14.28	Énumérer les systèmes maître DP d'un sous-réseau.....	203
7.14.29	Énumérer les connecteurs IO affectés.....	204
7.14.30	Relier IO-Connector DP à un réseau maître DP.....	205
7.15	Fonctions sur les appareils.....	206
7.15.1	Attributs obligatoires d'appareils.....	206
7.15.2	Appeler l'identifiant de type des appareils et des éléments d'appareil.....	207
7.15.3	Créer un appareil.....	210
7.15.4	Énumérer des appareils.....	211
7.15.5	Accéder à des appareils.....	214
7.15.6	Supprimer un appareil.....	216
7.16	Fonctions d'éléments d'appareils.....	218
7.16.1	Attributs obligatoires des éléments d'appareil.....	218
7.16.2	Créer et enficher un élément d'appareil.....	220
7.16.3	Déplacer un élément d'appareil dans un autre emplacement d'enfichage.....	223
7.16.4	Copier l'élément d'appareil.....	224
7.16.5	Supprimer un élément d'appareil.....	225
7.16.6	Énumérer des éléments d'appareil.....	226
7.16.7	Appeler des éléments d'appareil.....	227
7.16.8	Accès à l'élément d'appareil en tant qu'interface.....	231
7.16.9	Accéder aux attributs d'une interface d'appareil I/O.....	232
7.16.10	Accès aux attributs de l'loController.....	234

7.16.11	Accès aux attributs de l'IoConnector.....	235
7.16.12	Accéder à un contrôleur d'adresse.....	238
7.16.13	Accéder à des adresses.....	239
7.16.14	Accéder à l'"identifiant de matériel".....	241
7.16.15	Accéder au contrôleur d'identifiant de matériel.....	241
7.16.16	Accéder aux voies d'éléments d'appareil.....	242
7.17	Fonctions sur les données d'un appareil HMI.....	244
7.17.1	Vues.....	244
7.17.1.1	Créer des dossiers de vues personnalisés.....	244
7.17.1.2	Supprimer la vue d'un dossier.....	244
7.17.1.3	Supprimer un modèle de vue d'un dossier.....	245
7.17.1.4	Supprimer toutes les vues d'un dossier.....	246
7.17.2	Cycles.....	247
7.17.2.1	Suppression de cycle.....	247
7.17.3	Listes de textes.....	247
7.17.3.1	Suppression de la liste de textes.....	247
7.17.4	Listes de graphiques.....	248
7.17.4.1	Suppression d'une liste de graphiques.....	248
7.17.5	Connexions.....	249
7.17.5.1	Suppression de la liaison.....	249
7.17.6	Table des variables.....	249
7.17.6.1	Générer des dossiers personnalisés pour variables IHM.....	249
7.17.6.2	Enumérer les variables d'une table de variables IHM.....	250
7.17.6.3	Suppression de variables individuelles d'une table de variables IHM.....	251
7.17.6.4	Supprimer une table de variables d'un dossier.....	251
7.17.7	Scripts VB.....	252
7.17.7.1	Créer des dossiers personnalisés pour les scripts.....	252
7.17.7.2	Supprimer les scripts VB d'un dossier.....	253
7.17.8	Supprimer le dossier personnalisé d'un pupitre opérateur	253
7.18	Fonctions sur les données d'un appareil API.....	254
7.18.1	Déterminer le statut d'un API.....	254
7.18.2	Accéder aux paramètres d'une liaison en ligne.....	255
7.18.3	Fonctions de chargement de données dans l'API.....	259
7.18.3.1	Charger des composants matériels et logiciels dans l'API.....	259
7.18.3.2	Démarrage et arrêt d'un API.....	269
7.18.3.3	Prise en charge de rappels.....	270
7.18.3.4	Protection par mot de passe du PLC.....	271
7.18.3.5	Manipulation des mots de passe d'API liée aux blocs.....	273
7.18.4	Comparer le logiciel de l'API.....	274
7.18.5	Etablir ou interrompre une liaison en ligne à l'API.....	276
7.18.6	Blocs.....	278
7.18.6.1	Interroger le groupe "Blocs de programme".....	278
7.18.6.2	Enumérer les groupes Blocs personnalisés.....	278
7.18.6.3	Enumérer tous les blocs.....	279
7.18.6.4	Interroger les informations d'un bloc/type de données utilisateur.....	280
7.18.6.5	Supprimer un bloc.....	282
7.18.6.6	Créer un groupe pour blocs.....	282
7.18.6.7	Accéder aux attributs de tous les blocs.....	283
7.18.6.8	Créer un FB ProDiag.....	284
7.18.6.9	Accéder aux surveillances et aux propriétés du FB ProDiag.....	285
7.18.6.10	Supprimer un groupe pour blocs.....	286

7.18.6.11	Interroger un groupe système pour blocs système.....	287
7.18.6.12	Enumérer les sous-groupes système.....	287
7.18.6.13	Ajouter un fichier externe.....	288
7.18.6.14	Générer une source à partir d'un bloc.....	289
7.18.6.15	Générer les blocs à partir de la source.....	291
7.18.6.16	Supprimer un type de données utilisateur.....	292
7.18.6.17	Supprimer un fichier externe.....	293
7.18.6.18	Démarrer un éditeur de bloc.....	294
7.18.7	Objets technologiques.....	294
7.18.7.1	Vue d'ensemble des objets technologiques.....	294
7.18.7.2	Vue d'ensemble des objets technologiques et des versions.....	295
7.18.7.3	Vue d'ensemble des types de données.....	297
7.18.7.4	Interroger la composition des objets technologiques.....	298
7.18.7.5	Créer un objet technologique.....	298
7.18.7.6	Supprimer des objets technologiques.....	299
7.18.7.7	Compiler un objet technologique.....	300
7.18.7.8	Enumérer des objets technologiques.....	301
7.18.7.9	Rechercher un objet technologique.....	302
7.18.7.10	Enumérer les paramètres d'un objet technologique.....	302
7.18.7.11	Rechercher les paramètres d'un objet technologique.....	303
7.18.7.12	Lire les paramètres d'un objet technologique.....	304
7.18.7.13	Ecrire les paramètres d'un objet technologique.....	305
7.18.7.14	S7-1200 Motion Control.....	306
7.18.7.15	S7-1500 Motion Control.....	314
7.18.7.16	Régulation PID.....	333
7.18.7.17	Comptage.....	334
7.18.7.18	Easy Motion Control.....	334
7.18.8	Variables et tables de variables.....	335
7.18.8.1	Démarrage de l'éditeur de variables API.....	335
7.18.8.2	Interroger un groupe système pour variables API.....	336
7.18.8.3	Créer une table des variables API.....	336
7.18.8.4	Enumérer les groupes personnalisés pour variables API.....	337
7.18.8.5	Créer les groupes personnalisés pour variables API.....	338
7.18.8.6	Supprimer les groupes personnalisés pour variables API.....	339
7.18.8.7	Enumérer des tables de variables API dans un dossier.....	339
7.18.8.8	Interroger les informations d'une table de variables API.....	340
7.18.8.9	Lire la date et l'heure de la dernière modification d'une table de variables API.....	342
7.18.8.10	Supprimer la table des variables API dans un groupe.....	342
7.18.8.11	Enumérer des variables API.....	343
7.18.8.12	Accéder à des variables API.....	344
7.18.8.13	Accéder à des constantes API.....	345
7.19	Startdrive.....	348
7.19.1	Référence.....	348
7.19.1.1	DriveObject.....	348
7.19.1.2	OnlineDriveObject.....	348
7.19.1.3	DriveObjectContainer.....	349
7.19.1.4	OnlineDriveObjectContainer.....	349
7.19.1.5	DriveParameterComposition.....	350
7.19.1.6	DriveParameter.....	351
7.19.1.7	TelegramComposition.....	353
7.19.1.8	Telegram.....	354
7.19.1.9	TelegramType.....	355

7.19.1.10	AddressComposition.....	356
7.19.1.11	AddressContext.....	356
7.19.1.12	AddressIoType.....	357
7.19.1.13	StartDriveDownloadCheckConfiguration.....	357
7.19.2	Exemple de code.....	358
7.19.2.1	Créer un groupe d'entraînement.....	358
7.19.2.2	Créer un composant d'entraînement.....	359
7.19.2.3	Créer un composant pour un composant d'entraînement (uniquement S120).....	359
7.19.2.4	Déterminer un objet entraînement.....	361
7.19.2.5	Lire et écrire des paramètres.....	361
7.19.2.6	Lire et écrire des paramètres en ligne.....	362
7.19.2.7	Lire et écrire des paramètres FCOM.....	363
7.19.2.8	Insérer et étendre des télégrammes.....	364
7.19.2.9	Téléchargement.....	364
7.20	Exceptions.....	367
7.20.1	Traitement des exceptions.....	367
8	Exportation/importation.....	371
8.1	Vue d'ensemble.....	371
8.1.1	Notions élémentaires sur l'importation/exportation.....	371
8.1.2	Domaine d'utilisation de l'importation/exportation.....	373
8.1.3	Importation SimaticML spécifique à la version.....	374
8.1.4	Edition du fichier XML.....	375
8.1.5	Exportation de données de configuration.....	375
8.1.6	Importation de données de configuration.....	377
8.2	Importation/exportation de données du projet.....	379
8.2.1	Bibliothèque de graphiques.....	379
8.2.1.1	Exportation/importation de graphiques.....	379
8.2.1.2	Exporter les graphiques d'un projet.....	380
8.2.1.3	Importer des graphiques dans un projet.....	381
8.2.2	Textes du projet.....	382
8.2.2.1	Exportation de textes de projet.....	382
8.2.2.2	Importation de textes de projet.....	383
8.3	Importation/exportation de données d'un appareil IHM.....	385
8.3.1	Structure d'un fichier XML.....	385
8.3.2	Structure des données pour l'importation/exportation.....	387
8.3.3	Cycles.....	390
8.3.3.1	Exportation de cycles.....	390
8.3.3.2	Importer des cycles.....	391
8.3.4	Table des variables.....	392
8.3.4.1	Exporter des tables de variables IHM.....	392
8.3.4.2	Importer une table de variables IHM.....	395
8.3.4.3	Exporter des variables individuelles d'une table de variables IHM.....	396
8.3.4.4	Importer des variables individuelles d'une table de variables IHM.....	397
8.3.4.5	Particularités de l'importation/exportation de variables IHM.....	397
8.3.5	Scripts VB.....	399
8.3.5.1	Exporter des scripts VB.....	399
8.3.5.2	Exporter des scripts VB à partir d'un dossier.....	400
8.3.5.3	Importer des scripts VB.....	401
8.3.6	Listes de textes.....	402
8.3.6.1	Exporter des listes de textes à partir d'un appareil IHM.....	402

8.3.6.2	Importer une liste de texte dans un appareil IHM.....	403
8.3.6.3	Formats XML avancés pour l'exportation/importation de listes de textes.....	404
8.3.7	Listes de graphiques.....	406
8.3.7.1	Exporter les listes de graphiques.....	406
8.3.7.2	Importer les listes de graphiques.....	406
8.3.8	Connexions.....	407
8.3.8.1	Exporter des connexions.....	407
8.3.8.2	Importation de connexions.....	408
8.3.9	Vues.....	409
8.3.9.1	Vue d'ensemble des objets graphiques pouvant être exportés.....	409
8.3.9.2	Exporter toutes les vues d'un appareil IHM.....	413
8.3.9.3	Exporter une vue à partir d'un dossier de vues.....	414
8.3.9.4	Importer des vues dans un appareil IHM.....	416
8.3.9.5	Exporter une fenêtre permanente.....	418
8.3.9.6	Importer une fenêtre permanente.....	419
8.3.9.7	Exporter tous les modèles de vue d'un appareil IHM.....	420
8.3.9.8	Exporter des modèles de vue à partir d'un dossier.....	421
8.3.9.9	Importer des modèles de vue.....	423
8.3.9.10	Exportation d'une vue contextuelle.....	425
8.3.9.11	Importation d'une vue contextuelle.....	426
8.3.9.12	Exportation d'une vue encastrable.....	427
8.3.9.13	Importation d'une vue encastrable.....	429
8.3.9.14	Exportation d'une vue avec masque de saisie.....	430
8.3.9.15	Importation d'une vue avec masque de saisie.....	432
8.4	Importation/exportation de données d'un appareil API.....	436
8.4.1	Blocs.....	436
8.4.1.1	Structure XML de la section interface du bloc	436
8.4.1.2	Modifications du modèle d'objet et format de fichier XML.....	446
8.4.1.3	Exporter des blocs	448
8.4.1.4	Exporter des blocs avec protection know-how.....	454
8.4.1.5	Exportation/importation de blocs SCL.....	455
8.4.1.6	Exportation/importation de types structurés de blocs SCL.....	468
8.4.1.7	Exportation/importation de blocs d'appel SCL.....	474
8.4.1.8	Exporter des blocs F.....	491
8.4.1.9	Exporter des blocs système.....	491
8.4.1.10	Exporter des blocs GRAPH avec texte multilingue.....	492
8.4.1.11	Importer un bloc.....	493
8.4.2	Tables des variables.....	495
8.4.2.1	Exporter des tables de variables API.....	495
8.4.2.2	Importer une table de variables API.....	496
8.4.2.3	Exporter des variables ou constantes individuelles d'une table de variables API.....	497
8.4.2.4	Importer une seule variable ou constante dans une table de variables API.....	498
8.4.3	Exporter un type de données utilisateur.....	499
8.4.4	Importer un type de données utilisateur.....	500
8.4.5	Exportation de données au format OPC UA XML.....	501
8.5	Importation/exportation de données matérielles.....	503
8.5.1	Format de fichier AML.....	503
8.5.2	Pruned AML.....	503
8.5.3	Vue d'ensemble des objets et paramètres de l'importation/exportation CAx.....	505
8.5.4	Structure des données CAx pour l'importation/exportation.....	507
8.5.5	Identifiants de type AML.....	512

8.5.6	Exportation de données CAx.....	515
8.5.7	Importation de données CAx.....	519
8.5.8	Exception pour l'importation et exportation de données CAx.....	521
8.5.9	Appareils et modules Round-Trip.....	522
8.5.10	Topologie de l'exportation/importation.....	525
8.5.11	Exportation d'un élément d'appareil.....	527
8.5.12	Importation d'un objet d'appareil.....	529
8.5.13	Exportation/importation d'un appareil avec une adresse définie.....	532
8.5.14	Exportation/importation d'un appareil avec des voies.....	535
8.5.15	Exportation d'objets d'élément d'appareil.....	537
8.5.16	Importation d'objets d'élément d'appareil.....	541
8.5.17	Exportation/importation d'appareils et éléments d'appareils basés sur GSD/GSDML.....	544
8.5.18	Exportation/importation de sous-réseaux.....	549
8.5.19	Exportation/importation de variables API.....	556
8.5.20	Exportation/importation de réseaux IO.....	558
8.5.21	Exportation/importation de commentaires multilingues.....	560
8.5.22	Attributs AML comparés aux attributs TIA Portal Openness.....	562
9	Principales modifications.....	565
9.1	Modifications importantes dans V14 SP1.....	565
9.1.1	Modifications importantes dans V14 SP1.....	565
9.1.2	Principales modifications dans le modèle d'objet.....	568
9.1.3	Modifications apportées à la fonction du pilote.....	572
9.1.4	Modifications de l'importation et de l'exportation.....	577
9.1.4.1	Modifications de l'importation et de l'exportation.....	577
9.1.4.2	Modifications dans l'API.....	577
9.1.4.3	Extension des schémas.....	578
9.1.4.4	Modifications apportées au schéma.....	581
9.1.4.5	Modifications du comportement.....	584
9.1.4.6	Modifications apportées aux attributs de bloc.....	595
9.2	Modifications importantes dans V14.....	598
9.2.1	Principales modifications du modèle d'objet.....	598
9.2.2	Avant la mise à niveau d'une application vers TIA Portal Openness V14.....	600
9.2.3	Principales modifications de chaîne de caractères.....	601
9.2.4	Importation de fichiers créés avec TIA Portal Openness V13 SP1 et des versions antérieures.....	604
	Index.....	607

Consignes de sécurité

Informations de sécurité

Siemens commercialise des produits et solutions comprenant des fonctions de sécurité industrielle qui contribuent à une exploitation sûre des installations, systèmes, machines, équipements et/ou réseaux.

Pour protéger les installations, les systèmes, les machines et les réseaux des cybermenaces, il est nécessaire d'intégrer un concept de sécurité industrielle moderne complet et d'en assurer la maintenance. Les produits et solutions de Siemens ne constituent qu'une partie d'un tel concept.

Il incombe au client d'empêcher tout accès non autorisé à ses installations, systèmes, machines et réseaux. Les systèmes, machines et composants doivent uniquement être connectés au réseau d'entreprise ou à Internet dans la mesure où c'est nécessaire et en appliquant des mesures de protection correspondantes (p. ex. utilisation de pare-feux et segmentation du réseau).

En outre, il convient de respecter les directives de Siemens relatives aux mesures de sécurité correspondantes. Pour plus d'informations sur la sécurité industrielle, rendez-vous sur

<http://www.siemens.com/industrialsecurity> (<http://www.industry.siemens.com/topics/global/en/industrial-security/Pages/Default.aspx>)

Les produits et solutions Siemens font l'objet de développements continus pour être plus sûrs. Siemens recommande d'utiliser impérativement les mises à jour des produits dès qu'elles sont disponibles ainsi que les versions de produits les plus récentes. L'utilisation de versions de produits qui ne sont plus supportées et le non-respect des mises à jour les plus récentes peut augmenter le risque d'exposition du client à des cybermenaces.

Afin d'être informé des mises à jour des produits, veuillez souscrire au flux RSS Siemens Industrial Security à l'adresse

<http://www.siemens.com/industrialsecurity> (<http://www.industry.siemens.com/topics/global/en/industrial-security/Pages/Default.aspx>)

Lisezmoi TIA Portal Openness

2.1 Lisezmoi

Mesures de sécurité pour applications TIA Portal Openness

Il est recommandé

- d'installer une application TIA Portal Openness avec des droits d'administrateur dans le dossier "Programmes".
- d'éviter le chargement dynamique d'éléments de programme tels que des Assemblies ou DLL.
- D'exécuter l'application TIA Portal Openness avec des droits d'utilisateur.

Copie d'une application TIA Portal Openness

Lorsque vous copiez une application TIA Portal Openness exécutable, il est possible dans certaines circonstances que le chemin de répertoire dans lequel l'application TIA Portal Openness a été créée à l'origine soit lu par cette même application.

Solution :

Lorsque vous avez copié l'application TIA Portal Openness dans un nouveau répertoire, ouvrez par exemple la boîte de dialogue "Propriétés", puis refermez-la pour mettre à jour le cache de Windows.

Prise en charge de certaines fonctions dans un projet TIA Portal

Mode multi-utilisateurs

TIA Portal Openness ne prend en charge aucune opération administrative multi-utilisateurs. La raison est que l'utilisation de TIA Portal Openness n'est pas recommandée dans des projets multi-utilisateurs. Notez que certaines actions dans TIA Portal Openness peuvent même perturber le bon déroulement des opérations multi-utilisateurs définies par l'interface utilisateur de TIA Portal. Toutefois, si vous souhaitez effectuer des modifications avec TIA Portal Openness, exportez d'abord le projet multi-utilisateurs dans un projet à utilisateur unique.

Sécurité en cas de défaut

Si vous utilisez TIA Portal Openness, vous devez tenir compte de certaines restrictions concernant la sécurité en cas de défaut. Pour plus d'informations, voir la documentation "Systèmes de sécurité SIMATIC - Configuration et programmation".

Amélioration des performances de TIA Portal Openness

Pour atteindre le niveau de performance maximal de TIA Portal Openness, vous pouvez désactiver la fonction de recherche globale de TIA Portal. Pour désactiver la recherche globale, utilisez l'interface utilisateur ou l'appel de TIA Portal Openness API. À la fin du script TIA Portal Openness, vous pouvez de nouveau activer la recherche globale. La désactivation de la recherche globale permet certes d'améliorer le niveau de performance, mais toutes les fonctions TIA Portal Openness sont opérationnelles normalement même lorsque la recherche globale est activée.

Code du programme à thread sécurisé

Assurez-vous que votre code est à thread sécurisé : un événement apparaît dans différents threads.

Comportement d'exportation d'éléments d'écran avec style activé

Lors de l'exportation d'un élément d'écran avec style activé, les attributs de l'élément de style ne sont pas exportés, mais uniquement les attributs de l'élément d'écran avant l'activation du style. Lorsqu'un style est sélectionné et que "UseDesignColorSchema" est activé pour l'élément d'écran, ce dernier prend les valeurs des attributs du style sur l'interface utilisateur. Toutefois, dans la base de données, les attributs définis avant la sélection du style restent enregistrés pour cet élément d'écran. TIA Portal Openness exporte les valeurs réellement enregistrées dans la base de données.

Lorsque le style est désactivé puis réactivé et que l'élément d'écran est à nouveau exporté, ce sont les valeurs des attributs applicables dans l'élément de style qui sont exportées pour cet élément d'écran. Les valeurs d'attributs de l'élément de style sélectionné pour cet élément d'écran sont enregistrées dans la base de données lorsque "UseDesignColorSchema" n'est pas activé.

Ce problème peut être résolu de la manière suivante :

1. Affectez l'élément d'écran à l'élément de style :
 - La base de données contient les valeurs d'attributs valables avant l'activation du style.
 - L'interface utilisateur reprend les attributs directement de l'élément de style.
2. Exportez l'élément d'écran affecté à l'élément de style :
 - Le fichier XML contient les valeurs des attributs issues de la base de données, qui correspondent aux valeurs antérieures à l'activation du style.
3. Désactivez "UseDesignColorSchema" :
 - Les valeurs des attributs de l'élément de style sont ajoutées aux attributs de l'élément d'écran dans la base de données.

4. Activez "UseDesignColorSchema" :
 - Les valeurs des attributs de l'élément d'écran ne sont pas modifiées dans la base de données et correspondent toujours à celles de l'étape 3.
 - L'interface utilisateur reprend les attributs directement de l'élément de style.
5. Exportez l'élément d'écran affecté à l'élément de style :
 - Le fichier XML contient les valeurs des attributs issues de la base de données, qui ont été définies à l'étape 3 et qui correspondent aux valeurs de l'élément de style.

Copie des objets technologiques de S7-1500 pour la commande des mouvements

La copie de TO_CamTrack, TO_OutputCam ou TO_MeasuringInput à partir de la bibliothèque de projet ou de la bibliothèque globale dans le projet n'est pas possible.

Importation des esclaves ASi via AML

Si l'un des esclaves ASi suivants est importé par le biais d'un fichier AML alors la version du firmware de l'élément de l'appareil doit être placée dans tous les cas sur V13.0 :

- 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

Importation et exportation de touches de fonction

Les touches de fonction sont synchronisées lors de l'importation. Si une touche de fonction est créée dans un écran global et si la touche dans l'écran est vide, la touche de fonction correspondante utilise la définition globale dans tous les écrans.

Si vous voulez désactiver l'utilisation globale de touches de fonction après l'importation, définissez des touches vides dans les écrans et importez les types d'écrans dans l'ordre suivant : écran global, modèles, écrans.

Si, lors de l'exportation d'écrans, vous voulez vous assurer que la définition globale d'une touche de fonction n'est pas utilisée par le modèle ou par l'écran global, créez une touche de fonction vide dans l'écran. Sélectionnez les touches de fonction requises dans l'écran. Puis activez la propriété "Utiliser affectation globale" et désactivez-la de nouveau.

2.2 Modifications majeures dans TIA Portal Openness V15

Modifications

Si vous avez tenu compte des remarques concernant la programmation dans plusieurs versions et n'avez pas mis à jour votre projet à la version V15, vos applications fonctionnent sans aucune restriction sur tout ordinateur, même si seul TIA Portal V15 est installé.

Si vous mettez à jour votre projet à la version V15, il est nécessaire de recompiler votre application avec la SiemensEngineering.dll de version V15.

Dans certains cas, il est nécessaire d'adapter le code de votre application.

- Modifications du comportement pour les compositions dans DeviceItemComposition
- BitOffset d'adresses Asi
- Classe Exception
- Dossiers système de types de données utilisateur système
- Les sous-modules n'ont pas les attributs Author et TypeName
- Horodatage de la dernière modification
- Fichier XML d'exportation pour blocs GRAPH
- Importer des tables des variables
- Modifier les attributs pertinents non relatifs à la sécurité d'un API
- Modifier les paramètres de sécurité en cas de réglage d'un mot de passe de sécurité
- Accès à des objets dans une CPU S7-1200

Modifications du comportement pour les compositions dans DeviceItemComposition

Les compositions suivantes dans DeviceItemComposition ont été modifiées afin d'atteindre un comportement dynamique : la composition est maintenant mise à jour lorsqu'un élément est ajouté ou supprimé via l'interface utilisateur de TIA Portal.

- IoSystem – ConnectedIoDevices
- Subnet – IoSystems
- Subnet – Nodes
- NetworkInterface – Nodes
- NetworkInterface – Ports
- NetworkPort – ConnectedPorts
- SubnetOwner – Subnets

BitOffset d'adresses Asi

Si un module a une adresse d'entrée et une adresse de sortie pour les deux objets d'adresse, l'attribut BitOffset correct est fourni.

Si un module est doté de voies, l'attribut BitOffset n'est pas fourni pour la voie.

Classe Exception

ServiceID et MessageID ont été supprimés de la classe exception.

Les sous-modules n'ont pas les attributs Author et TypeName

Les attributs Author et TypeName ont été supprimés des sous-modules qui ne peuvent pas être enfichés.

Dossiers système de types de données utilisateur système

Pour les dossiers système de types de données utilisateur, le dossier correspondant et la composition correspondante sont mis à disposition. Cela conduit également à une modification dans la hiérarchie de résultats de comparaison.

Horodatage de la dernière modification

Si un objet est modifié pendant la mise à niveau d'un objet, l'horodatage de la dernière modification est également modifié.

Fichier XML d'exportation pour blocs GRAPH

Le fichier XML d'exportation pour des blocs GRAPH contient une action vide supplémentaire : <Actions />.

Importer des tables des variables

Le réglage des attributs de variable ne dépend plus du type de données.

Modifier les attributs pertinents non relatifs à la sécurité d'un API

Tous les attributs pertinents non relatifs à la sécurité d'un API peuvent être modifiés via TIA Portal Openness, même si un mot de passe de sécurité est paramétré.

Modifier les paramètres de sécurité en cas de réglage d'un mot de passe de sécurité

Les paramètres de sécurité d'E/S de sécurité ne peuvent être modifiés que si le mot de passe de sécurité n'est pas paramétré.

Accès à des objets dans une CPU S7-1200

L'accès aux variables de tableau pour les objets technologiques TO_PositioningAxis et TO_CommandTable a été modifié. Vous trouverez des informations à ce sujet au chapitre concernant S7-1200 Motion Control.

2.3 Notification des modifications majeures dans les prochaines versions

Notification des modifications

La TIA Portal Openness API sera modifié dans une version ultérieure. Il n'y a pas besoin de changer immédiatement le code de votre application car les applications fonctionnent sans aucune restriction sur la base de versions antérieures. Mais, pour les nouvelles applications, il est recommandé d'utiliser les nouvelles fonctionnalités et de planifier le recodage votre application car, à partir de V17, les méthodes suivantes ne sont plus prises en charge.

- Type de compositions

Type de compositions

Les types suivants sont modifiés afin d'afficher le comportement d'image de couple :

- AddressAssociation
- AddressComposition
- AddressControllerAssociation
- ChannelComposition
- DeviceItemAssociation
- DeviceItemComposition
- HwIdentifierAssociation
- HwIdentifierComposition
- HwIdentifierControllerAssociation
- IoConnectorComposition
- IoControllerComposition

2.4 Remarques sur l'écriture d'un code stable à long terme

Changement de version

Si vous tenez compte de quelques remarques concernant l'écriture d'un code stable à long terme, vous serez en mesure d'utiliser votre application avec d'autres versions de TIA Portal sans modifier son code.

Chemin d'accès au registre et fichier appconfig

Des modifications sont nécessaires pour modifier le chemin d'accès au registre et le fichier appconfig. Exemple :

"C:\Programmes\Siemens\Automation\Portal V14\PublicAPI\V14
SP1\Siemens.Engineering.dll"

doit être changé en

"C:\Programmes\Siemens\Automation\Portal V15\PublicAPI\V14
SP1\Siemens.Engineering.dll".

Pour écrire un code stable à long terme, le chemin d'accès au registre doit être configurable et le fichier appconfig doit être mis à jour.

Chemin d'installation

Des modifications sont nécessaires pour modifier le chemin d'installation de TIA Portal.

Exemple :

"C:\Programmes\Siemens\Automation\Portal V14\PublicAPI\V14
SP1\Siemens.Engineering.dll"

doit être changé en

"C:\Programmes\Siemens\Automation\Portal V15\PublicAPI\V14
SP1\Siemens.Engineering.dll".

Pour écrire un code stable à long terme, le chemin d'installation doit être configurable.

Chemin d'accès de AmiHost

Des modifications sont nécessaires pour modifier le chemin d'accès à AmiHost. Exemple :

"C:\Programmes\Siemens\Automation\Portal V14\bin

\Siemens.Automation.Cax.AmiHost.exe"

doit être changé en "C:\Programmes\Siemens\Automation\Portal V15\bin

\Siemens.Automation.Cax.AmiHost.exe".

Pour écrire un code stable à long terme, le chemin d'accès à AmiHost doit être configurable.

Extensions de fichiers de projet et bibliothèques de TIA Portal

Des modifications sont nécessaires pour modifier les extensions des fichiers de projet et des bibliothèques de TIA Portal. Exemple :

*.ap14

doit être changé en

*.ap15.

2.4 Remarques sur l'écriture d'un code stable à long terme

Pour écrire un code stable à long terme, les extensions des fichiers de projet et des bibliothèques de TIA Portal doivent être configurables.

Ouverture d'un projet

Pour écrire un code stable à long terme, la méthode `Projects.OpenWithUpgrade` doit être utilisée au lieu de la méthode `Projects.Open`.

Hiérarchie lors de la comparaison, la compilation ou le téléchargement de résultats

La hiérarchie et/ou l'ordre lors de la comparaison, de la compilation ou du téléchargement de résultats peut changer selon la version.

Pour écrire un code stable à long terme, vous devez éviter d'émettre des hypothèses quant à la profondeur et à l'ordre de certains résultats.

Nouveautés de TIA Portal Openness

Nouveau modèle d'objet TIA Portal Openness

Les nouvelles fonctions et innovations suivantes sont disponibles dans TIA Portal Openness V15. Vous trouverez des détails sur les thèmes abordés dans les différentes rubriques de la documentation produit.

- Les DLL Openness de V14SP1 et V15 sont incluses dans la fourniture
Comme les DLL Openness de V14SP1 et V15 sont contenues dans la fourniture, les applications basées sur V14 SP1 fonctionnent également sans modification dans V15. Pour utiliser les fonctions de la version V15, vous devez intégrer les DLL de V15 et compiler de nouveau l'application.
- Exportation et importation de blocs SCL
Il est possible d'exporter et d'importer au format XML des blocs SCL ainsi que des blocs CONT ou LOG comprenant des réseaux SCL.
- Chargement d'API
Le chargement d'API S7-1500 standard peut être automatisé. Pour cela, l'API est arrêté et démarré de manière implicite. Les mots de passe de niveau de protection et les mots de passe associés peuvent être transmis par l'application.
- Lecture d'un total de contrôle
Le total de contrôle du programme API standard peut être lu par un API hors ligne.
- ProDiag
Des FB, des instances et des affectations peuvent être créées pour ProDIAG.
- Dossiers système pour UDT
Les UDT système sont accessibles dans le groupe système pour des types de données utilisateur.
- Numérotation des blocs
La numérotation automatique des blocs peut être activée et désactivée. Les numéros de bloc peuvent également être modifiés.
- Objets technologiques
Openness contient des extensions pour le téléchargement et l'enregistrement de données du traceur TO et pour les nouvelles fonctionnalités TO.
- Startdrive
Il est possible de créer des objets DriveObjects et des trames pour les entraînements SINAMICS G120 et SINAMICS S120. Des paramètres d'entraînement sélectionnés peuvent être définis en ligne et hors ligne. Le chargement dans l'appareil est possible.

Pour plus d'informations sur les modifications du modèle d'objet, référez-vous à AUTOHOTSPOT.

Voir aussi

Modèle d'objet TIA Portal Openness (Page 51)

Modifications importantes dans V14 SP1 (Page 565)

Modifications importantes dans V14 (Page 598)

notions de base

4.1 Conditions requises pour TIA Portal Openness

Conditions préalables à l'utilisation d'applications TIA Openness

- Un produit basé sur TIA Portal est installé sur le PC, tel que "STEP 7 Professional" ou "WinCC Professional".
- TIA Openness est installé sur le PC.
Voir Installation de TIA Openness (Page 25)

Systèmes d'exploitation Windows pris en charge

Le tableau suivant montre les combinaisons du système d'exploitation Windows, de TIA Portal et de l'application utilisateur qui sont compatibles :

Système d'exploitation Windows	TIA Portal	Application utilisateur
64 bits	64 bits	32 bits, 64 bits et AnyCPU

Conditions préalables à la programmation d'applications TIA Portal Openness

- Microsoft Visual Studio 2015 Update 1 ou une version supérieure avec .Net 4.6.2

Savoir-faire nécessaire de l'utilisateur

- Connaissances en ingénierie système
- Connaissances avancées de Microsoft Visual Studio 2015 Update 1 ou d'une version supérieure avec .Net 4.6.2
- Connaissances avancées de C# / VB.net et .Net
- Connaissances sur l'utilisation de TIA Portal

Canaux Remoting TIA Portal Openness

Les canaux Remoting TIA Portal Openness sont enregistrés comme type IpcChannel, le paramètre "ensureSecurity" étant mis sur "false".

Remarque

Evitez d'enregistrer un autre IpcChannel avec le paramètre "ensureSecurity" différent de "false" et une priorité supérieure ou égale à "1".

4.1 Conditions requises pour TIA Portal Openness

Les attributs suivants sont définis pour l'IpcChannel :

Attribut	Paramètres
"name" et "portName"	Mis sur "\${Process.Name}_{Process.Id}" ou "\${Process.Name}_{Process.Id}_{AppDomain.Id}" si enregistré dans un autre AppDomain que l'AppDomain par défaut de l'application.
"priority"	Mis à la valeur par défaut "1".
"typeFilterLevel"	Mis sur "Complet".
"authorizedGroup"	Mis sur la chaîne de caractères de la valeur de compte NT pour le compte d'utilisateur intégré (c.-à-d. Tous)

Voir aussi

Ajouter un utilisateur au groupe d'utilisateurs "Siemens TIA Openness" (Page 26)

4.2 Installation

4.2.1 Installation de TIA Openness

Introduction

TIA Openness est automatiquement installé par le programme d'installation de TIA Portal. L'utilisateur doit s'assurer que la case à cocher pour TIA Openness (sous Options) est activée lors de l'installation de TIA Portal. Le fichier d'installation est une archive à décompression automatique se trouvant sur le DVD du produit.

Conditions

- Le matériel et les logiciels de l'appareil de programmation ou du PC sont conformes à la configuration système requise.
- Vous détenez les droits d'administrateur.
- Les programmes en cours d'exécution ont été fermés.
- La fonction d'exécution automatique est désactivée.
- WinCC et/ou STEP 7 est/sont installé(s).
- Le numéro de version de "TIA Portal Openness" correspond aux numéros de version de WinCC et STEP 7.

Remarque

Lorsqu'une version antérieure de TIA Openness est déjà installée, la version actuelle est installée parallèlement.

Marche à suivre

Pour installer TIA Openness, assurez-vous que la case à cocher pour TIA Openness est activée pendant l'installation de TIA Portal. Procédez comme suit pour contrôler l'installation de TIA Openness.

1. Dans le menu "Configuration", sélectionnez le répertoire "Options".
2. Cochez la case pour TIA Openness.
3. Cliquez sur "Next" puis sélectionnez l'option requise.

Suivez la procédure d'installation de TIA Portal pour compléter l'installation de TIA Openness.

Résultat

TIA Openness est installé sur le PC. En outre, le groupe d'utilisateurs local "Siemens TIA Openness" est créé.

Remarque

Le complément "TIA Openness" ne suffit pas pour continuer d'avoir accès à TIA Portal. Vous devez être membre du groupe d'utilisateurs "Siemens TIA Openness" (voir Ajouter un utilisateur au groupe d'utilisateurs "Siemens TIA Openness" (Page 26)).

4.2.2 Ajouter un utilisateur au groupe d'utilisateurs "Siemens TIA Openness"

Introduction

Lorsque vous installez TIA Portal Openness sur le PC, le groupe d'utilisateurs "Siemens TIA Openness" est automatiquement créé.

Chaque fois que vous accédez à TIA Portal avec votre application TIA Portal Openness, TIA Portal vérifie si vous êtes membre du groupe d'utilisateurs "Siemens TIA Openness", soit directement ou indirectement via un autre groupe d'utilisateurs. Si vous êtes membre du groupe d'utilisateurs "Siemens TIA Openness", l'application TIA Portal Openness démarre et établit une liaison à TIA Portal.

Marche à suivre

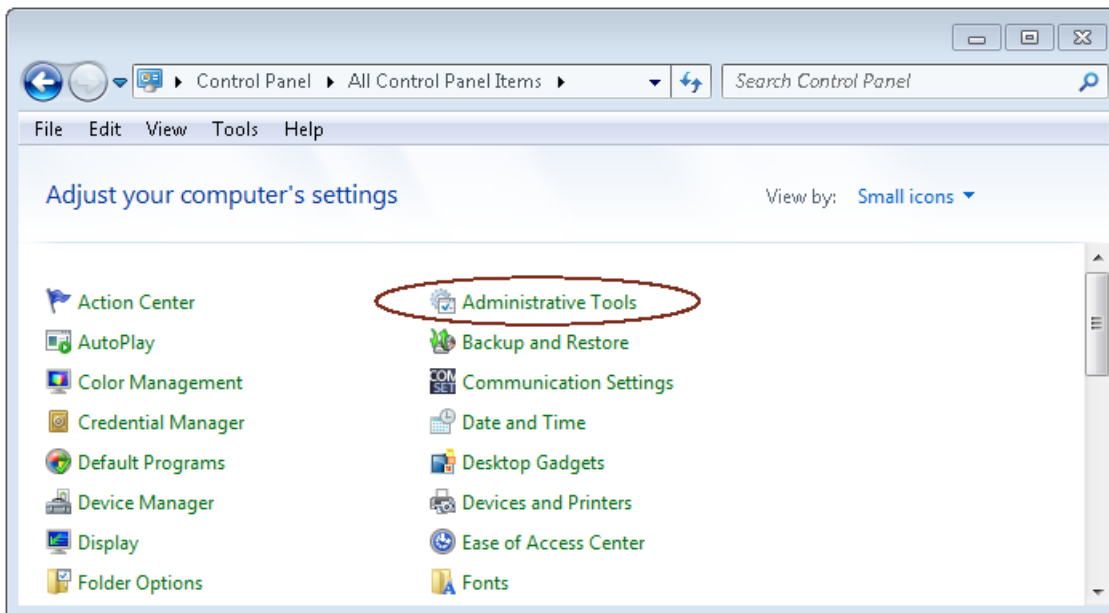
Vous ajoutez un utilisateur au groupe d'utilisateurs "Siemens TIA Openness" grâce à des applications de votre système d'exploitation. Le portail TIA ne prend pas en charge ce processus.

Remarque

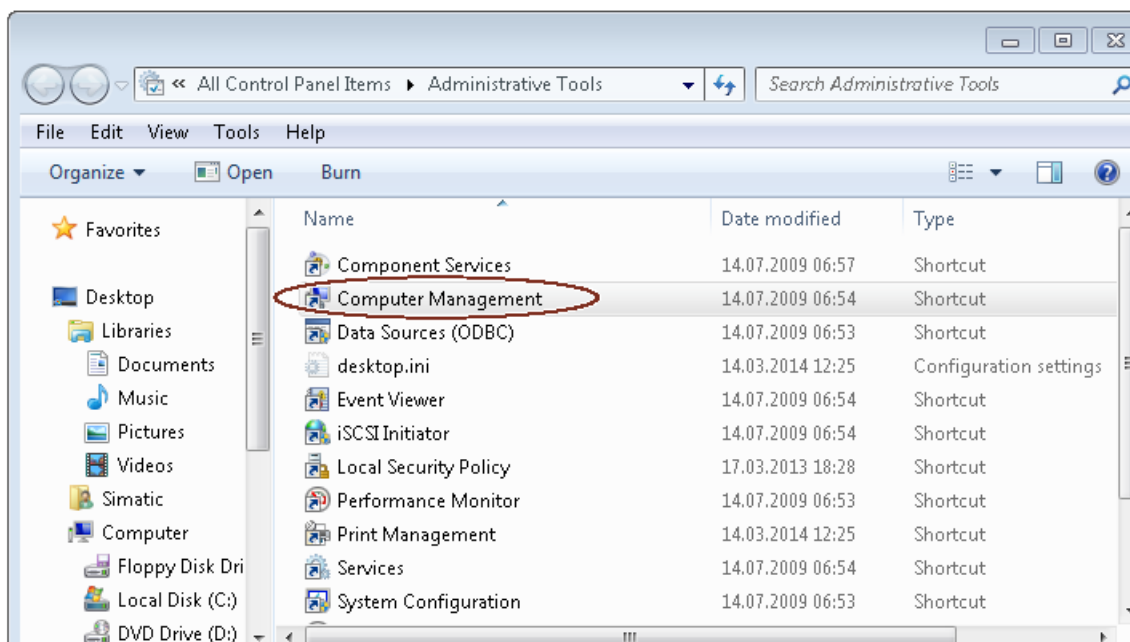
En fonction de la configuration de votre domaine ou de votre PC, les droits d'administrateur sont requis pour l'extension d'un groupe d'utilisateurs.

Avec le système d'exploitation Windows 7 (avec l'anglais comme langue paramétrée), vous pouvez par exemple ajouter un utilisateur au groupe d'utilisateurs comme suit :

1. Sélectionnez "Start" > "Control Panel".
2. Double-cliquez dans le panneau de configuration sur "Administrative Tools".



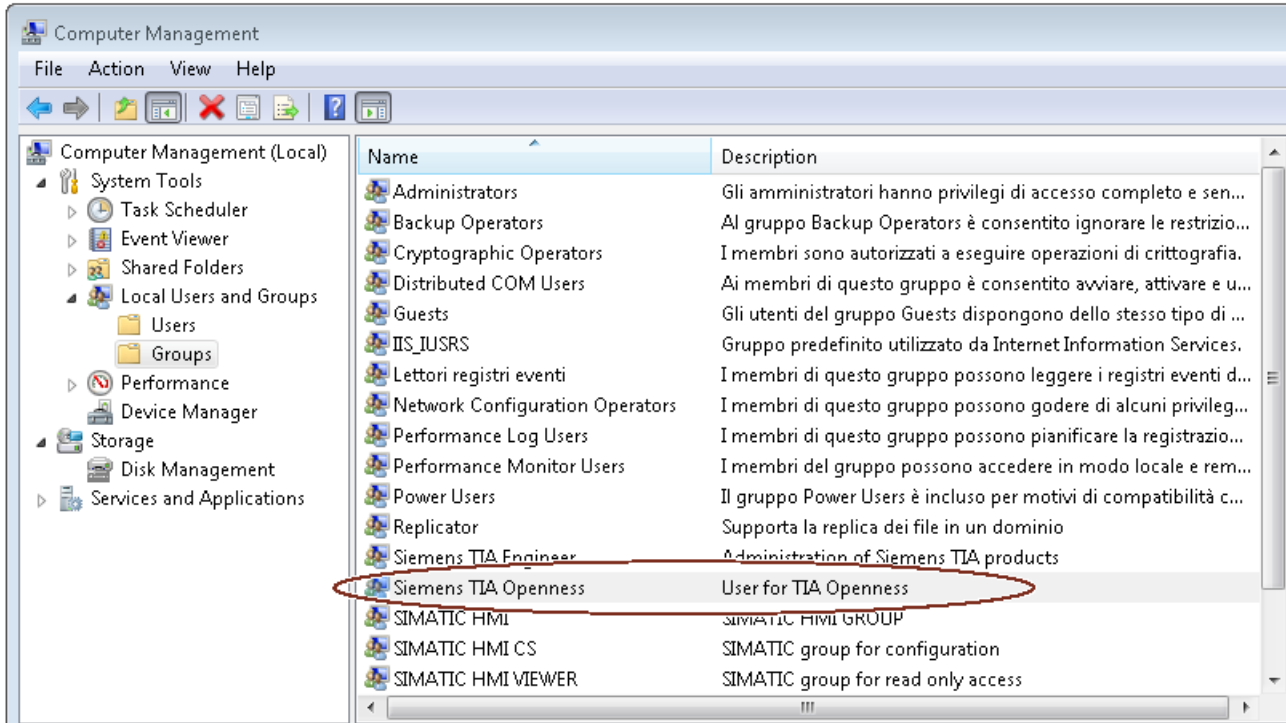
3. Cliquez sur "Computer Management" pour ouvrir la boîte de dialogue de configuration du même nom.



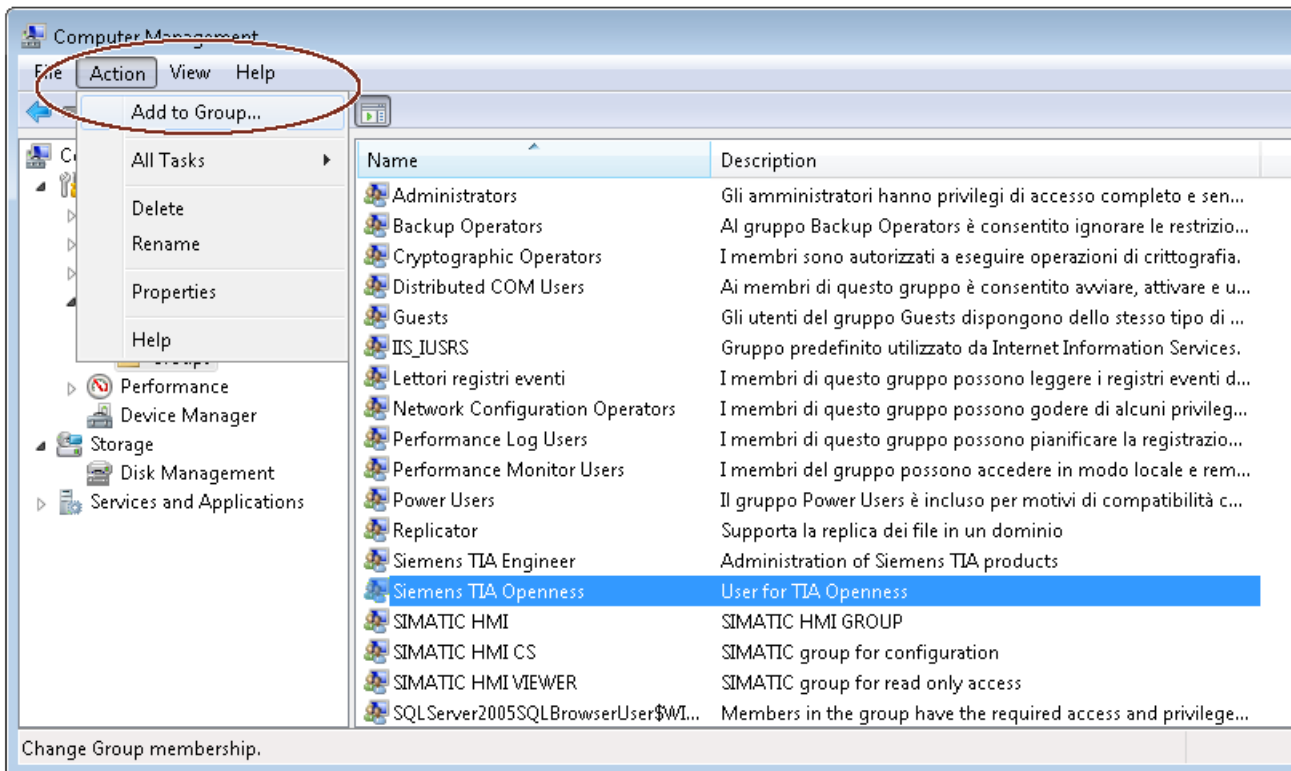
4. Sélectionnez "Local Users and Groups > Groups" pour afficher tous les groupes d'utilisateurs créés.

4.2 Installation

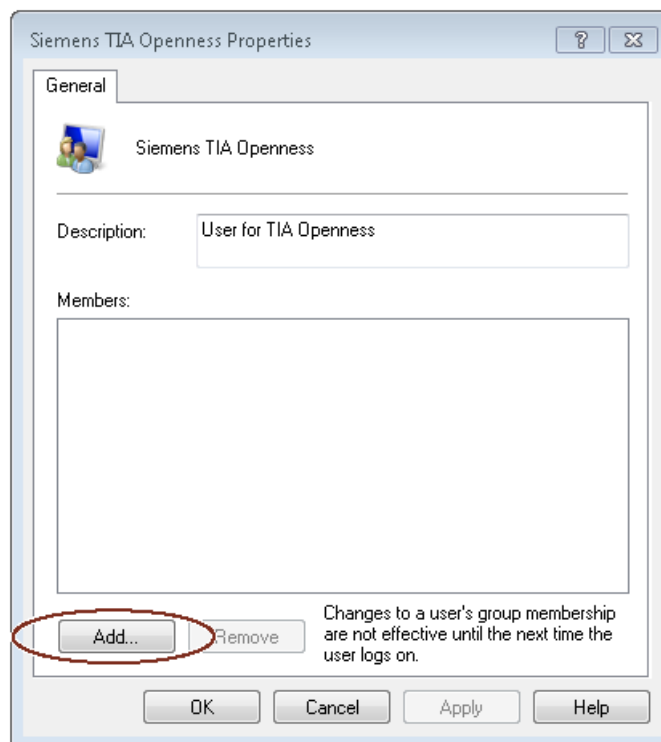
- 5. Dans la liste des groupes d'utilisateurs, sélectionnez dans le volet à droite l'entrée "Siemens TIA Openness".



6. Sélectionnez la commande de menu "Action > Add to Group...".

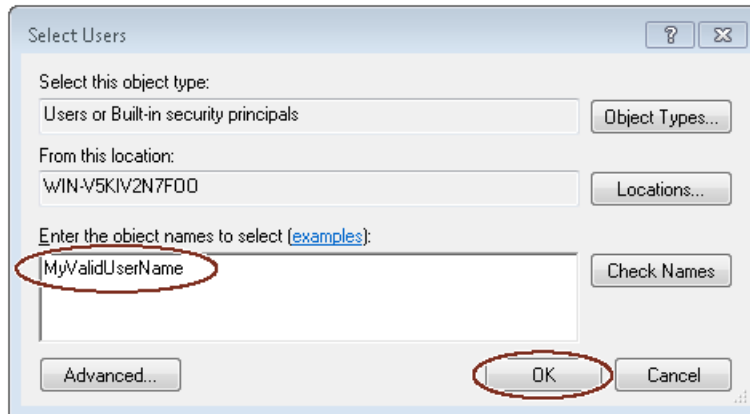


La boîte de dialogue des attributs du groupe d'utilisateurs s'ouvre :



7. Cliquez sur "Add".

Une boîte de dialogue de sélection regroupant les utilisateurs pouvant être sélectionnés s'ouvre alors :



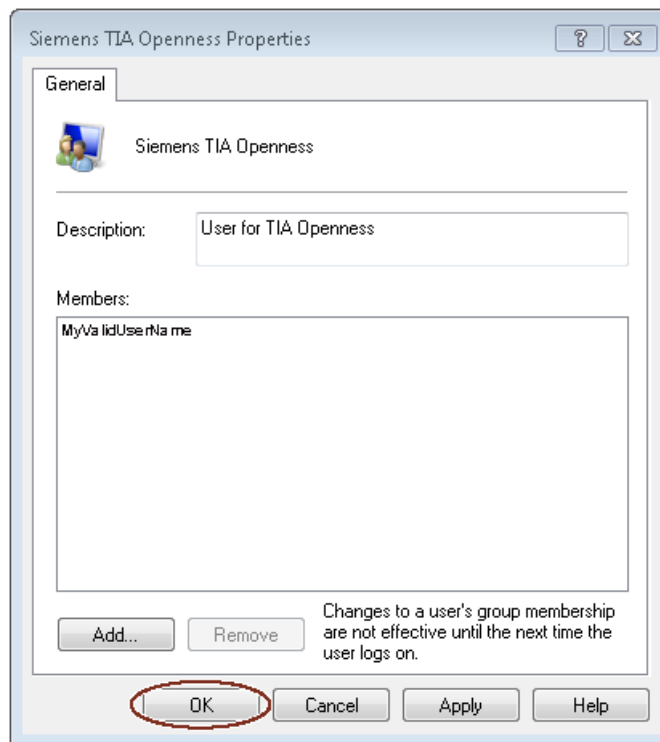
8. Entrez un nom d'utilisateur valide dans le champ de saisie.

Remarque

Cliquez sur le bouton "Check Names" pour vérifier si l'utilisateur saisi dispose d'un compte utilisateur valable pour ce domaine ou cet ordinateur.

Le champ "From this location" affiche le domaine ou le nom de l'ordinateur du nom d'utilisateur saisi. Vous pouvez obtenir des informations supplémentaires auprès de votre administrateur système.

9. Confirmez votre sélection par "OK".
Le nouvel utilisateur s'affiche alors dans la boîte de dialogue des attributs du groupe d'utilisateurs.



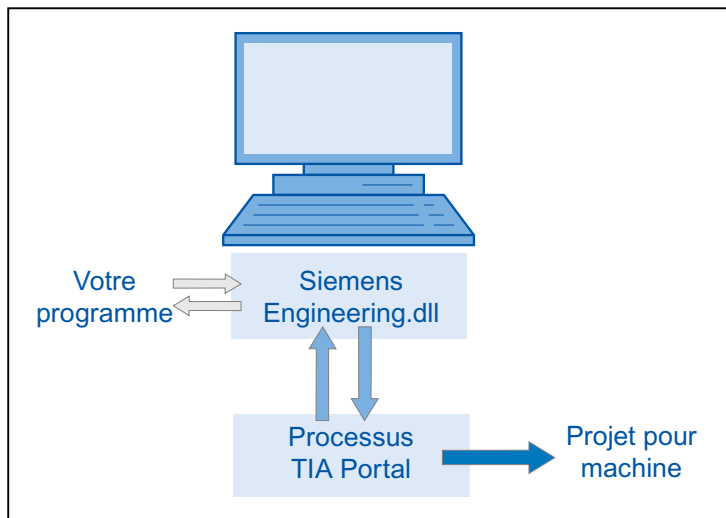
Vous saisissez d'autres utilisateurs à l'aide du bouton "Add".

10. Cliquez sur "OK" pour mettre fin à cette opération.
11. Connectez-vous de nouveau au PC pour que les modifications deviennent effectives.

4.2.3 Accéder au portail TIA

Vue d'ensemble

Ordinateur de configuration



Marche à suivre

1. Pour accéder et lancer TIA Portal, configurez votre environnement de développement.
2. Dans votre programme, créez une instance de l'objet de l'application de TIA Portal pour démarrer ce dernier.
3. Cherchez le projet souhaité et ouvrez-le.
4. Accédez aux données de projet.
5. Fermez le projet et quittez le portail TIA.

Voir aussi

Etablissement d'une connexion au portail TIA (Page 74)

Mettre fin à la connexion au portail TIA (Page 84)

4.3 Tâches d'Openness

4.3.1 Possibilités d'utilisation

Introduction

TIA Portal Openness vous offre différentes possibilités d'accès à TIA Portal et une sélection de fonctions pour des tâches définies.

Vous accédez aux zones suivantes de TIA Portal par le biais de l'interface TIA Portal Openness API :

- Données du projet
- Données API
- Données IHM

Remarque

L'interface TIA Portal Openness API ne doit pas servir à procéder à des contrôles ou à créer des données utilisées pour la réception/validation d'une installation de sécurité. Seuls une impression de sécurité réalisée avec le progiciel STEP 7 Safety ou le test fonctionnel conviennent pour une réception/validation. La TIA Portal Openness API ne saurait les remplacer.

Accéder à TIA Portal

Avec TIA Portal Openness , vous disposez de différentes possibilités d'accès à TIA Portal. Ce faisant, vous créez une instance externe de TIA Portal dans le processus, avec ou sans interface utilisateur. Vous pouvez également accéder en parallèle à des processus en cours de TIA Portal.

Accéder aux projets et données de projet

Lorsque vous accédez à des projets et des données de projet, vous utilisez TIA Portal Openness principalement pour les tâches suivantes :

- Ouvrir, fermer et enregistrer un projet
- Enumérer et interroger des objets
- Créer des objets
- Supprimer des objets

4.3.2 Exportation/importation

Introduction

TIA Portal Openness prend en charge l'importation et l'exportation de données de projet au moyen de fichiers XML. La fonction d'importation/exportation permet la configuration externe de données d'ingénierie existantes. Vous utilisez cette fonction pour rendre votre processus d'ingénierie plus efficace et éviter les erreurs.

Utilisation

Vous utilisez la fonction d'importation/exportation aux fins suivantes :

- Echange de données
- Copie de parties d'un projet
- Traitement externe de données de configuration, par exemple pour des opérations sur données de masse avec rechercher et remplacer
- Traitement externe de données de configuration pour de nouveaux projets sur la base de configurations existantes
- Importation de données de configuration générées en externe, telles que des listes de textes et des variables
- Mise à disposition de données de projet pour des applications externes

4.4 Liste d'objets

Introduction

Les tableaux suivants indiquent les objets disponibles, y compris Runtime Advanced et si ces objets sont pris en charge par TIA Portal Openness.

Ni le logiciel de visualisation Runtime Professional ni les fichiers de proxy d'appareil ne sont pris en charge par TIA Portal Openness dans WinCC.

Objets

Selon le pupitre opérateur que vous utilisez vous pouvez avoir recours aux données de projet suivantes :

Tableau 4-1 Vues

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Vue	oui	oui	oui	oui	oui	oui
Vue globale	oui	oui	oui	oui	oui	oui
Modèles	oui	oui	oui	oui	oui	oui
Fenêtre permanente	non	oui	oui	oui	oui	oui
Vue contextuelle	non	non	oui	non	oui	oui
Vue Slide-in	non	non	oui	non	oui	oui

Tableau 4-2 Objets de vue

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Ligne	oui	oui	oui	oui	oui	oui
Ligne polygonale	non	oui	oui	oui	oui	oui
Polygone	non	oui	oui	oui	oui	oui
Ellipse	oui	oui	oui	oui	oui	oui
Segment d'ellipse	non	non	non	non	non	non
Segment de cercle	non	non	non	non	non	non
Arc d'ellipse	non	non	non	non	non	non
Affichage caméra	non	non	non	non	non	non
Arc de cercle	non	non	non	non	non	non
Cercle	oui	oui	oui	oui	oui	oui
Affichage PDF	non	non	non	non	non	non
Rectangle	oui	oui	oui	oui	oui	oui
Connecteur	non	non	non	non	non	non

4.4 Liste d'objets

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Champ de texte	oui	oui	oui	oui	oui	oui
Affichage graphique	oui	oui	oui	oui	oui	oui
Tuyau	non	non	non	non	non	non
Double raccord en T	non	non	non	non	non	non
Raccord en T	non	non	non	non	non	non
Coude	non	non	non	non	non	non
Champ d'E/S	oui	oui	oui	oui	oui	oui
Champ date/heure	oui	oui	oui	oui	oui	oui
Champ d'E/S graphique	oui	oui	oui	oui	oui	oui
Champ de texte éditable	non	non	non	non	non	non
Champ de liste	non	non	non	non	non	non
Zone de liste déroulante	non	non	non	non	non	non
Bouton	oui	oui	oui	oui	oui	oui
Bouton rond	non	non	non	non	non	non
Bouton-poussoir lumineux	non	non	non	non	oui	non
Commutateur	oui	oui	oui	oui	oui	oui
Champ d'E/S symbolique	oui	oui	oui	oui	oui	oui
Commutateur à clé	non	non	non	non	oui	non
Bargraphe	oui	oui	oui	oui	oui	oui
Bibliothèque d'icônes	non	oui	oui	oui	oui	oui
Curseur	non	oui	oui	oui	oui	oui
Barre de défilement	non	non	non	non	non	non
Case à cocher	non	non	non	non	non	non
Bouton d'option	non	non	non	non	non	non
Instrument à aiguille	non	oui	oui	oui	oui	oui
Horloge	non	oui	oui	oui	oui	oui
Vue de l'espace mémoire	non	non	non	non	non	non
Touches de fonction	oui	oui	oui	oui	oui	oui
Instances de bloc d'affichage	non	non	oui	oui	oui	oui
Fenêtre de vues	non	non	non	non	non	non

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Vue des utilisateurs	oui	oui	oui	oui	oui	oui
Navigateur HTML	non	non	non	non	non	non
Travail d'impression/Diagnostic de script	non	non	non	non	non	non
Vue de recette	non	non	non	non	non	non
Vue des alarmes	non	non	non	non	non	non
Indicateur d'alarme	non	non	non	non	non	non
Fenêtre d'alarmes	non	non	non	non	non	non
Vue de courbes f(x)	non	non	non	non	non	non
Vue de courbes f(t)	non	non	non	non	non	non
Vue tabellaire	non	non	non	non	non	non
Table des valeurs	non	non	non	non	non	non
Media Player	non	non	non	non	non	non
Diagnostic de voie	non	non	non	non	non	non
WLAN - Réception	non	non	non	non	non	non
Zone - Nom	non	non	non	non	non	non
Zone - Signal	non	non	non	non	non	non
Nom de la plage d'action	non	non	non	non	non	non
Nom de la plage d'action (RFID)	non	non	non	non	non	non
Signal de la plage d'action	non	non	non	non	non	non
Etat de chargement	non	non	non	non	non	non
Molette	non	non	non	non	oui	non
Indicateur d'aide	non	non	non	non	non	non
Vue Sm@rtClient	non	non	non	non	non	non
Visualisation/forçage	non	non	non	non	non	non
Vue de diagnostic système	non	non	non	non	non	non
Fenêtre de diagnostic système	non	non	non	non	non	non

4.4 Liste d'objets

Tableau 4-3 Dynamiquement

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Affichage	oui	oui	oui	oui	oui	oui
Commande opérateur	non	oui	oui	oui	oui	oui
Visibilité	oui	oui	oui	oui	oui	oui
Déplacements	oui	oui	oui	oui	oui	oui

Tableau 4-4 Objets supplémentaires

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Groupes	oui	oui	oui	oui	oui	oui
Touches programmables	oui	oui	oui	oui	oui	oui
Cycles	oui	oui	oui	oui	oui	oui
Scripts VB	non	oui	oui	oui	oui	oui
Listes de fonctions	oui	oui	oui	oui	oui	oui
Bibliothèque de graphiques	oui	oui	oui	oui	oui	oui

Tableau 4-5 Variables

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Variables multiples	oui	oui	oui	oui	oui	oui
Tableaux	oui	oui	oui	oui	oui	oui
Types de données utilisateur	oui	oui	oui	oui	oui	oui
Interne	non	oui	oui	oui	oui	oui
Occurrences des types de données élémentaires	oui	oui	oui	oui	oui	oui
Occurrences de types de données utilisateur	oui	oui	oui	oui	oui	oui
Occurrence de tableaux	oui	oui	oui	oui	oui	oui

En outre, TIA Portal Openness prend en charge toutes les plages de valeurs prises en charge par les pilotes de communication.

Connexions

TIA Portal Openness prend en charge les connexions non intégrées également prises en charge par les pupitres opérateur correspondants. Vous trouverez des informations

complémentaires à ce sujet dans l'aide en ligne de TIA Portal sous "Visualisation de processus > Communication avec les API > Dépendance par rapport à l'appareil".

Tableau 4-6 Listes

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Listes de textes	oui	oui	oui	oui	oui	oui
Listes de graphiques	oui	oui	oui	oui	oui	oui

Tableau 4-7 Textes

Objet	Basic Panels	Panels	Comfort Panels	Multi Panels	Mobile Panels	RT Advanced
Textes multilingues	oui	oui	oui	oui	oui	oui
Textes formatés et leurs occurrences	non	oui	oui	oui	oui	oui

4.5 Bibliothèques standard

Pour que les exemples de code fonctionnent, ajoutez les instructions Namespace suivantes au début de l'exemple de code correspondant :

```
using System;
using Siemens.Engineering;
using Siemens.Engineering.CAx;
using Siemens.Engineering.HW;
using Siemens.Engineering.HW.Extension;
using Siemens.Engineering.HW.Features;
using Siemens.Engineering.HW.Utilities;
using Siemens.Engineering.SW;
using Siemens.Engineering.SW.Blocks;
using Siemens.Engineering.SW.TechnologicalObjects;
using Siemens.Engineering.SW.TechnologicalObjects.Motion;
using Siemens.Engineering.SW.ExternalSources;
using Siemens.Engineering.SW.Tags;
using Siemens.Engineering.SW.Types;
using Siemens.Engineering.Hmi;
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.Online;
using Siemens.Engineering.Compiler;
using Siemens.Engineering.Library;
using Siemens.Engineering.Library.Types;
using Siemens.Engineering.Library.MasterCopies;
using Siemens.Engineering.Compare;
using System.IO;
```


4.6 Remarques sur la performance de TIA Portal Openness

Objet racine

Vous pouvez indiquer plusieurs objets racine dans les fichiers d'importation.

Exemple : vous pouvez créer plusieurs listes de textes dans un fichier XML au lieu d'une liste de textes par fichier XML.

Fonctions de TIA Portal Openness

Lorsque vous ouvrez une fonction TIA Portal Openness pour la première fois, l'appel peut durer plus longtemps que les appels suivants de cette fonction TIA Portal Openness.

Exemple : si vous exécutez plusieurs exportations de données de configuration les unes après les autres, la première exportation peut durer plus longtemps que les suivantes.

Introduction

Introduction

TIA Portal Openness décrit des interfaces ouvertes pour l'ingénierie avec TIA Portal. Pour plus d'informations sur "TIA Portal Openness - Efficient generation of program code using code generators", voir la chaîne YouTube SIEMENS (www.youtube.com/watch?v=Ki12pLbEcxs).

Avec TIA Portal Openness, vous pouvez automatiser l'ingénierie en commandant à distance le TIA Portal à partir d'un programme créé par vos soins.

TIA Portal Openness vous permet d'exécuter les actions suivantes :

- Créer des données de projet
- Modifier des projets et des données de projet
- Supprimer des données de projet
- Lire des données de projet
- Mettre à disposition des projets et des données de projet pour d'autres applications.

Remarque

Siemens n'est pas responsable de la compatibilité des données transmises par le biais de ces interfaces avec un logiciel tiers et ne fournit aucune garantie à cet égard.

Nous attirons expressément l'attention sur le fait que l'utilisation inadéquate des interfaces peut entraîner la perte de données ou l'arrêt de la production.

Remarque

Les sections de code contenues dans cette documentation sont rédigées dans la syntaxe C#.

En raison de l'utilisation de sections de code courtes, une grande partie du traitement des erreurs n'est pas décrite.

Utilisation

Utilisez l'interface TIA Portal Openness aux fins suivantes :

- mettre à disposition des données de projet
- accéder au processus TIA Portal
- utiliser des données de projet

Utilisation de valeurs par défaut du domaine de l'automatisation

- par l'importation de données générées à distance
- par la commande à distance du portail TIA pour la génération de projets

Mise à disposition de données de projet du portail TIA pour des applications externes

- par l'exportation de données de projet

Conserver les atouts vis-à-vis de la concurrence par une ingénierie efficace

- Vous n'avez besoin de configurer les données d'ingénierie existantes dans le TIA Portal.
- Les processus d'ingénierie automatisés remplacent l'ingénierie manuelle.
- La réduction des coûts d'ingénierie renforce la position concurrentielle.

Travail en commun sur des données de projet

- Les tests de routine et le traitement groupé des données peuvent avoir lieu parallèlement à la configuration en cours.

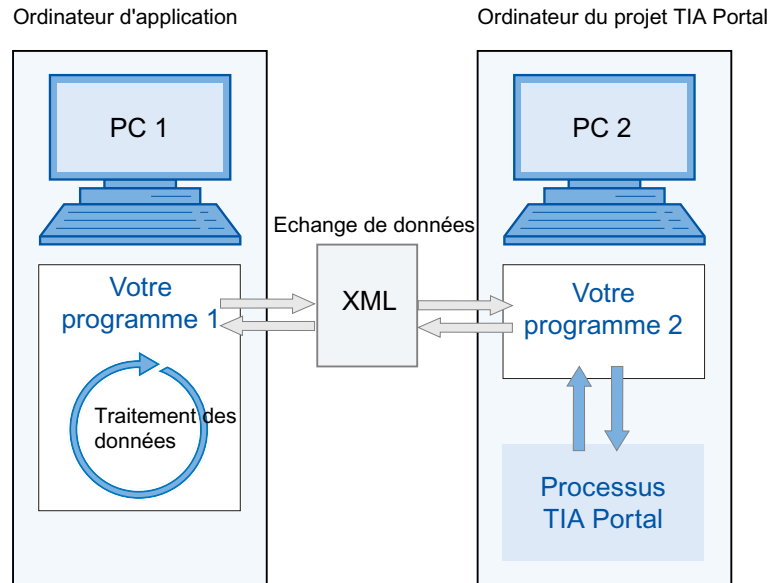
Voir aussi

Configurations (Page 45)

Configurations

Vous pouvez travailler avec deux variantes de TIA Portal Openness :

L'application et le TIA Portal se trouvent sur différents ordinateurs



- Les données sont échangées via des fichiers XML. Les fichiers XML peuvent être exportés ou importés par vos programmes.
- Les données exportées du portail TIA vers PC2 peuvent être modifiées sur PC1 et réimportées.

Remarque

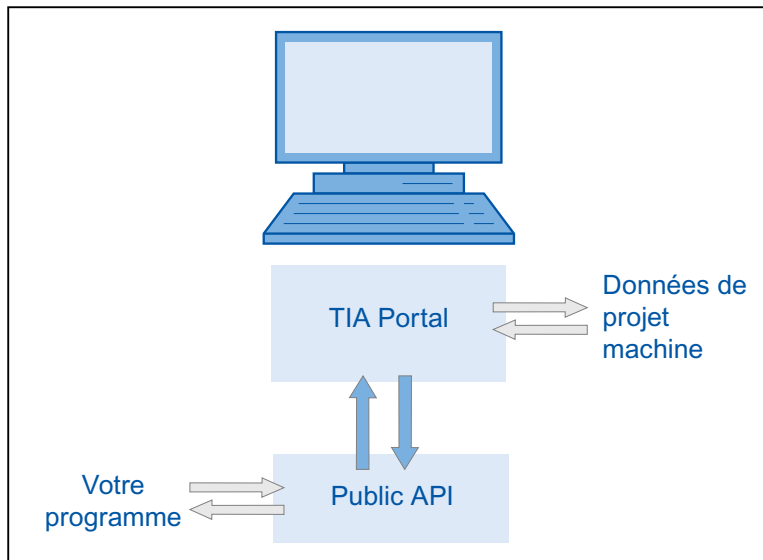
Vous devez développer un programme exécutable "Votre programme 2" pour PC2, par exemple "programm2.exe". Le portail TIA s'exécute avec ce programme en arrière-plan.

L'importation et l'exportation de fichiers XML s'effectuent exclusivement par le biais de la TIA Portal Openness API.

- Vous pouvez archiver les données échangées à des fins de vérification.
- Les données échangées peuvent être éditées à différents endroits et différents moments.

L'application et le TIA Portal se trouvent sur le même ordinateur

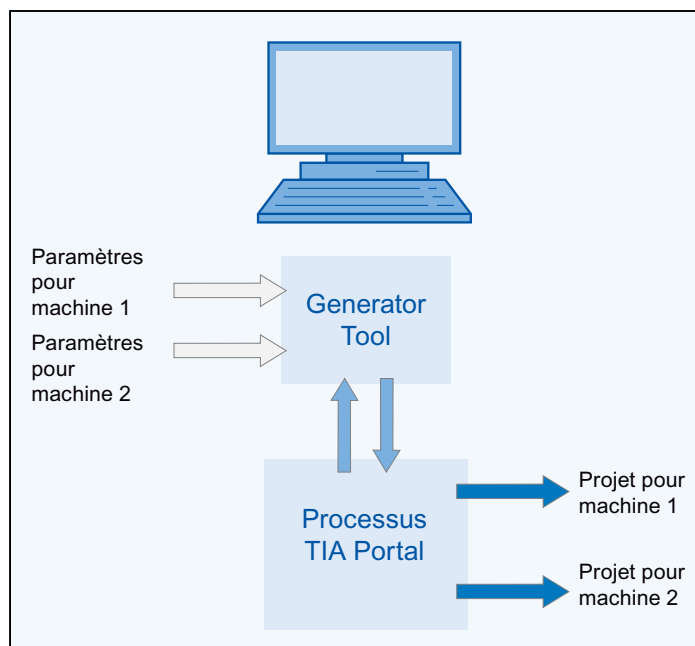
Ordinateur de configuration



- Votre programme lance le TIA Portal avec ou sans interface utilisateur. Votre programme ouvre, enregistre et/ou ferme un projet. Le programme peut également établir une liaison avec un TIA Portal en cours d'exécution.
- Vous pouvez alors utiliser les fonctionnalités du TIA Portal pour demander, générer et modifier des données de projet ou pour déclencher des processus d'importation et d'exportation.
- Les données sont créées sous le contrôle du traitement du portail TIA et enregistrées dans les données du projet.

Un domaine d'application typique est la construction de machines modulaire.

Ordinateur de configuration



- Un système d'automatisation efficace doit être appliqué sur des machines similaires.
- Un projet comprenant les composants de tous les modèles de machines est disponible dans le portail TIA.
- L'outil Generator Tool commande la création du projet correspondant à un modèle de machine donné.
- Il appelle les valeurs par défaut en lisant les paramètres du modèle de machine demandé.
- Il filtre les éléments concernés du projet global du portail TIA, les modifie le cas échéant et génère le projet de machine demandé.

TIA Portal Openness API

7.1 Introduction

Vue d'ensemble

TIA Portal Openness prend en charge une sélection de fonctions pour des tâches définies. Vous pouvez appeler ces fonctions par le biais de TIA Portal Openness API en-dehors de TIA Portal.

Remarque

Lorsqu'une version antérieure de TIA Portal Openness est déjà installée, la version actuelle est installée parallèlement.

Vous trouverez ci-après une vue d'ensemble des étapes de programmation typiques. Vous découvrirez comment les différentes sections de code interagissent et comment intégrer les différentes fonctions dans un programme complet. En outre, vous verrez quels éléments de code doivent être adaptés pour chaque tâche.

Exemple de programme

Les différentes étapes de programmation sont expliquées à l'aide de l'exemple de fonction "Créer l'accès de l'API dans une application console". Dans ce code de programme, vous intégrez les fonctions mises à disposition et vous adaptez les éléments de code correspondants pour cette tâche.

Fonctions

La rubrique ci-dessous indique les fonctions pour des tâches définies que vous pouvez appeler avec TIA Portal Openness en-dehors de TIA Portal.

Voir aussi

Possibilités d'utilisation (Page 33)

Liste d'objets (Page 35)

7.2 Etapes de programmation

Vue d'ensemble

TIA Portal Openness requiert les étapes de programmation suivantes pour l'accès via la TIA Portal Openness API :

1. Faire connaître TIA Portal dans l'environnement de développement
2. Configurer l'accès du programme à TIA Portal
3. Activer l'accès du programme à TIA Portal
4. Publier et démarrer TIA Portal
5. Ouvrir un projet
6. Exécuter des commandes
7. Enregistrer et fermer un projet
8. Mettre fin à la connexion au portail TIA

Remarque

Chaînes de caractères autorisées

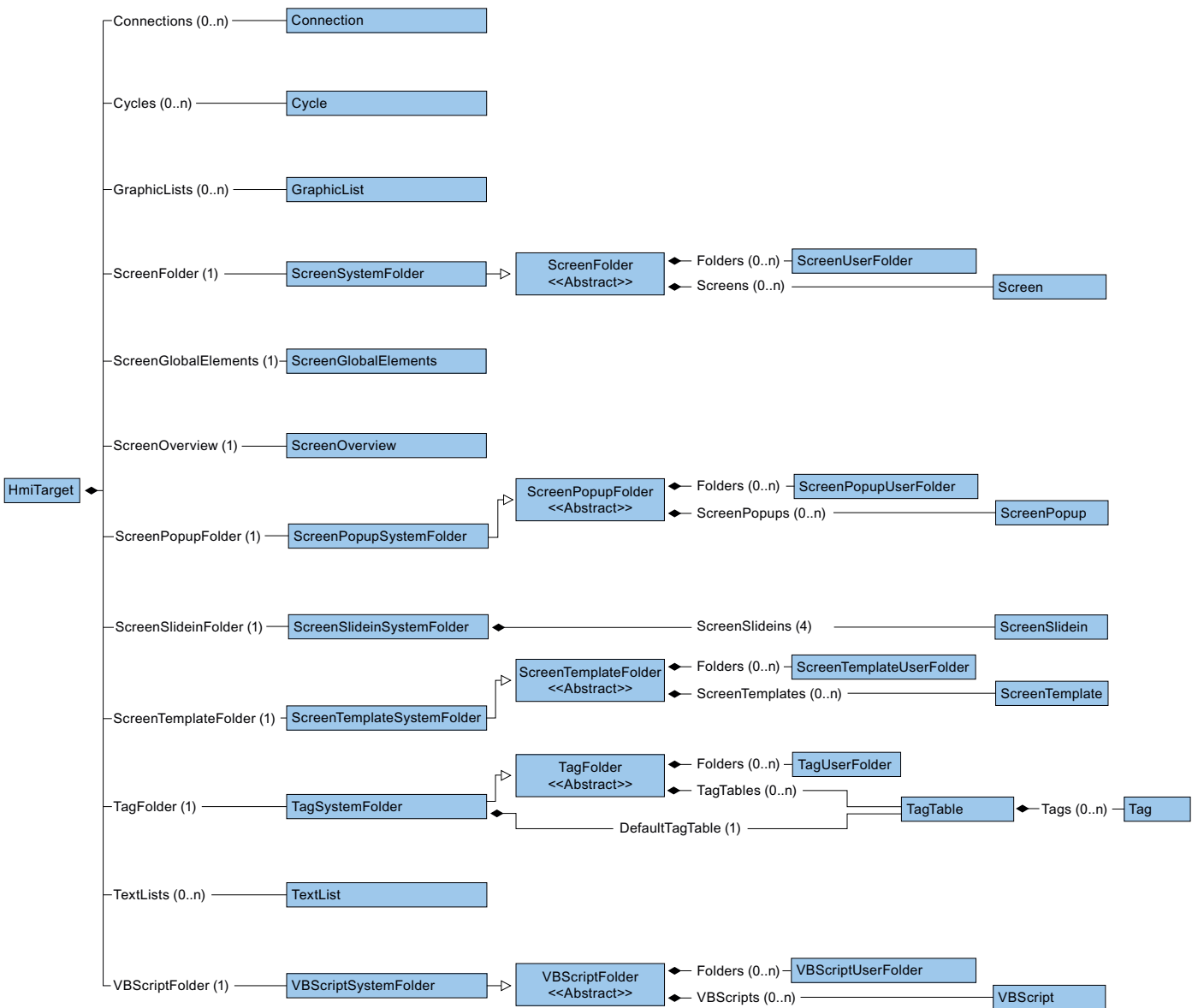
Seuls certains caractères sont autorisés dans les chaînes de caractères sur TIA Portal. Toutes les chaînes de caractères transmises à TIA Portal par le biais de l'application TIA Portal Openness doivent se conformer à ces règles. Si vous transmettez un caractère non autorisé dans un paramètre, le système déclenche une exception.

Voir aussi

Exemple de programme (Page 67)

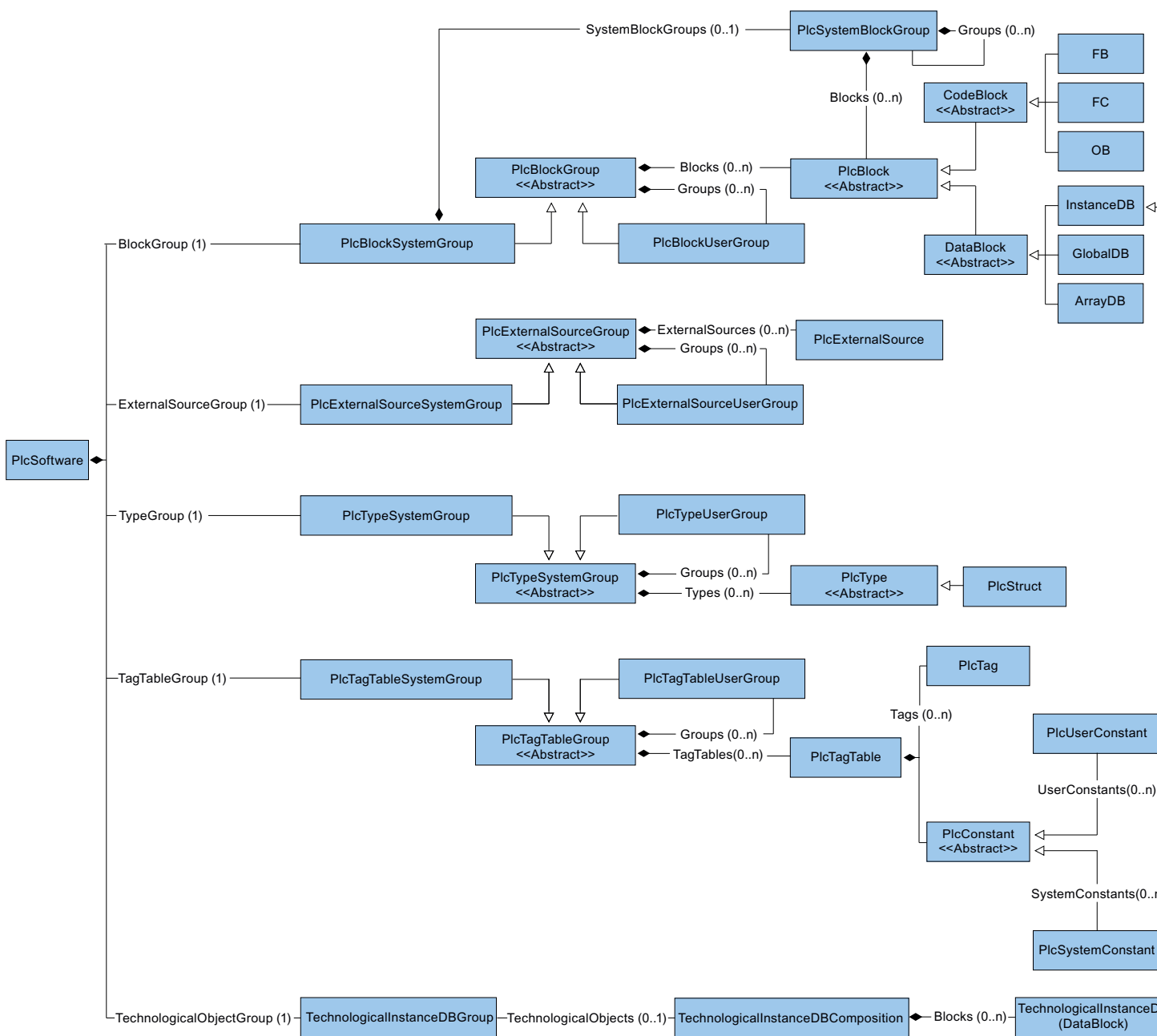
Utilisation des exemples de code (Page 72)

7.3 Modèle d'objet TIA Portal Openness



Le schéma suivant illustre les objets disponibles sous PlcSoftware.

7.3 Modèle d'objet TIA Portal Openness



Accéder à des objets de listes

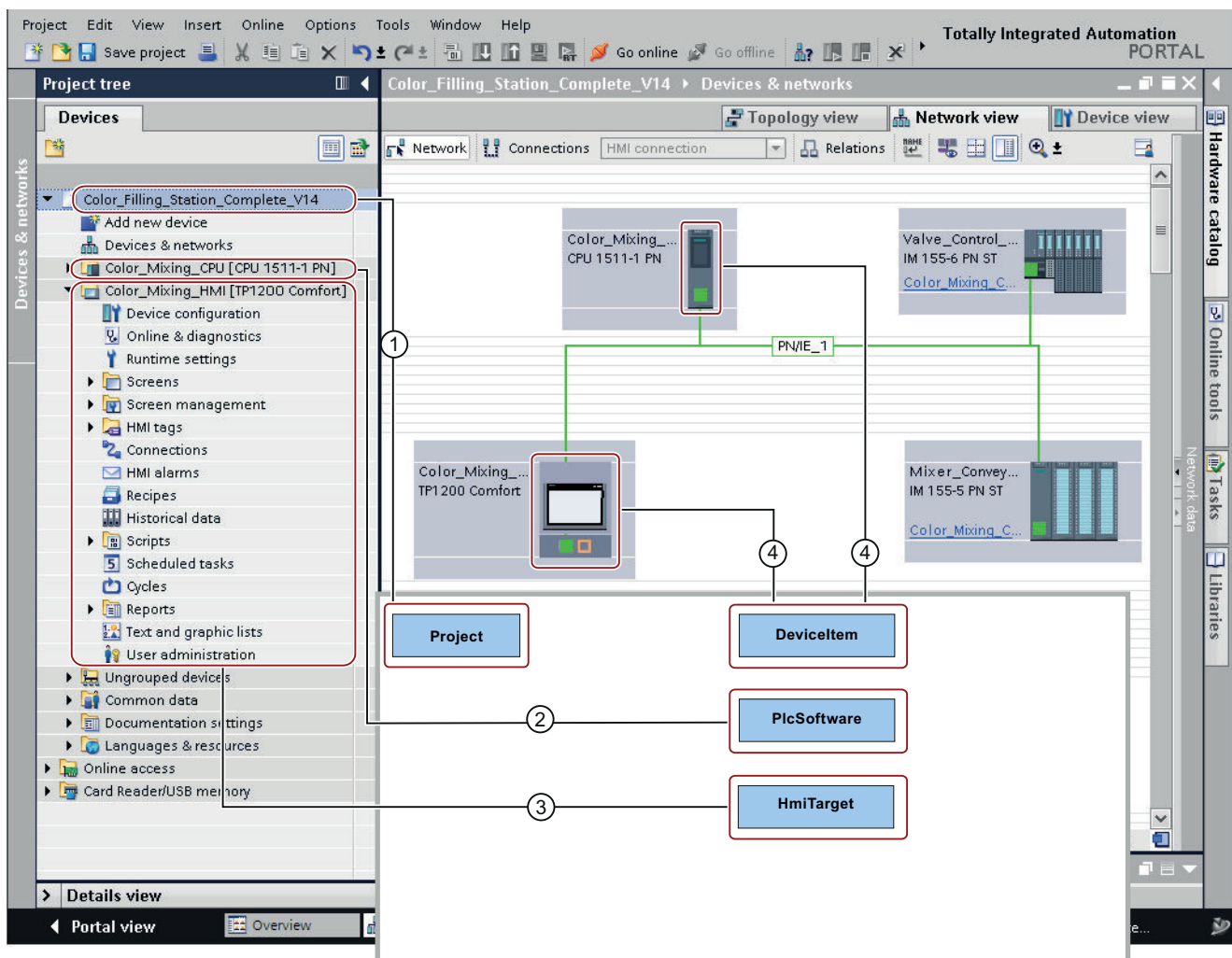
Pour adresser un objet dans une liste, vous avez les options suivantes :

- Adressez via l'index. Le comptage au sein des listes commence à partir de 0.
- Utilisez la méthode `Find`.
Utilisez cette méthode pour adresser un objet par son nom. Vous pouvez appliquer cette méthode à une composition ou une liste. La méthode `Find` n'est pas récursive.
Exemple :

```
ScreenComposition screens = folder.Screens;
Screen screen = screens.Find("myScreen");
```
- Utilisez les noms symboliques.

Relation entre TIA Portal et le modèle d'objet TIA Portal Openness

La figure suivante présente la relation entre le modèle d'objet et un projet dans TIA Portal :



Voir aussi

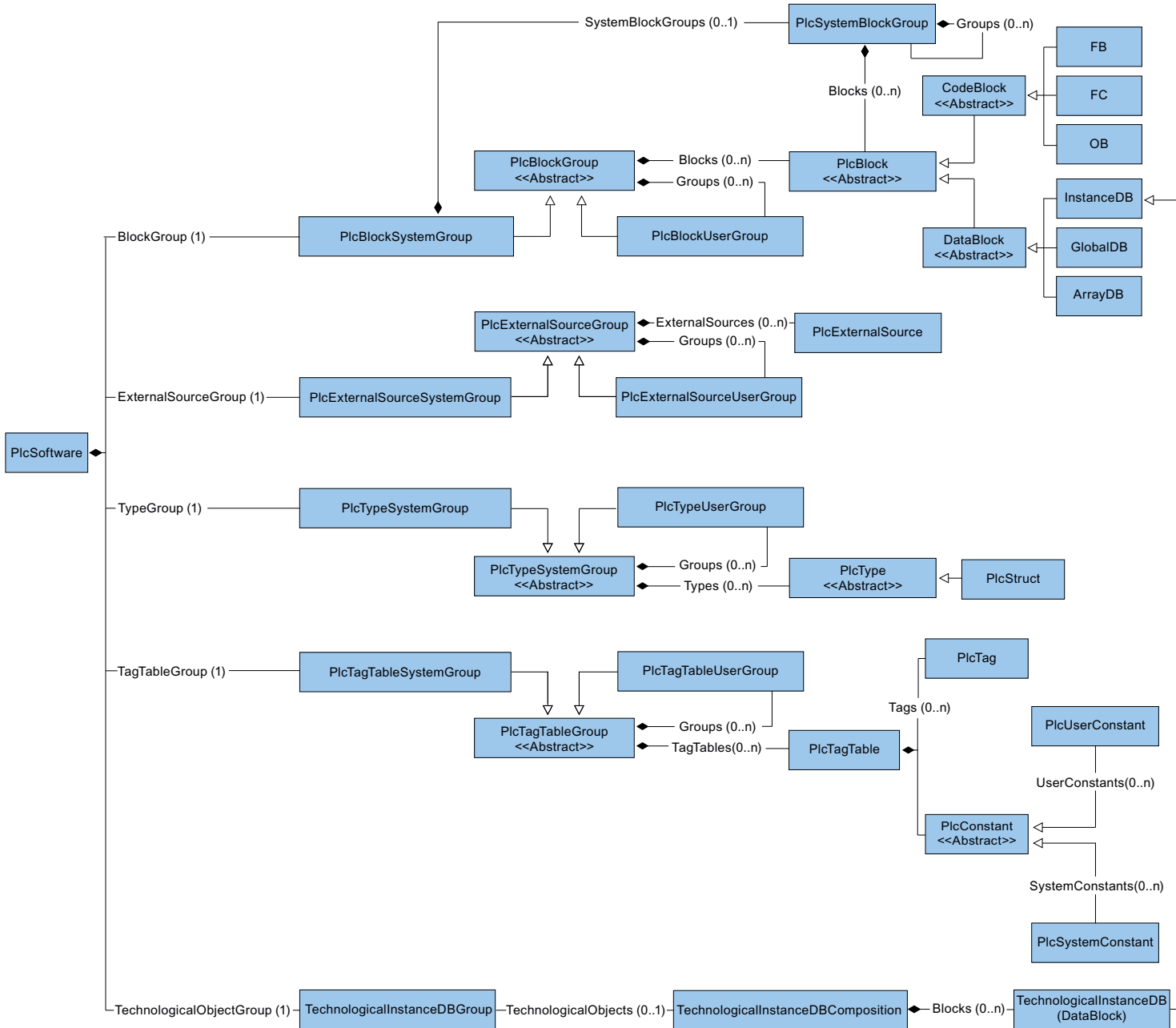
Blocs et types de modèle d'objet TIA Portal Openness (Page 56)

Hiérarchie des objets matériels du modèle d'objet (Page 64)

7.4 Blocs et types de modèle d'objet TIA Portal Openness

Introduction

Le schéma suivant représente le modèle de domaine des API afin de donner une vue d'ensemble de la structuration actuelle dans TIA Portal Openness.



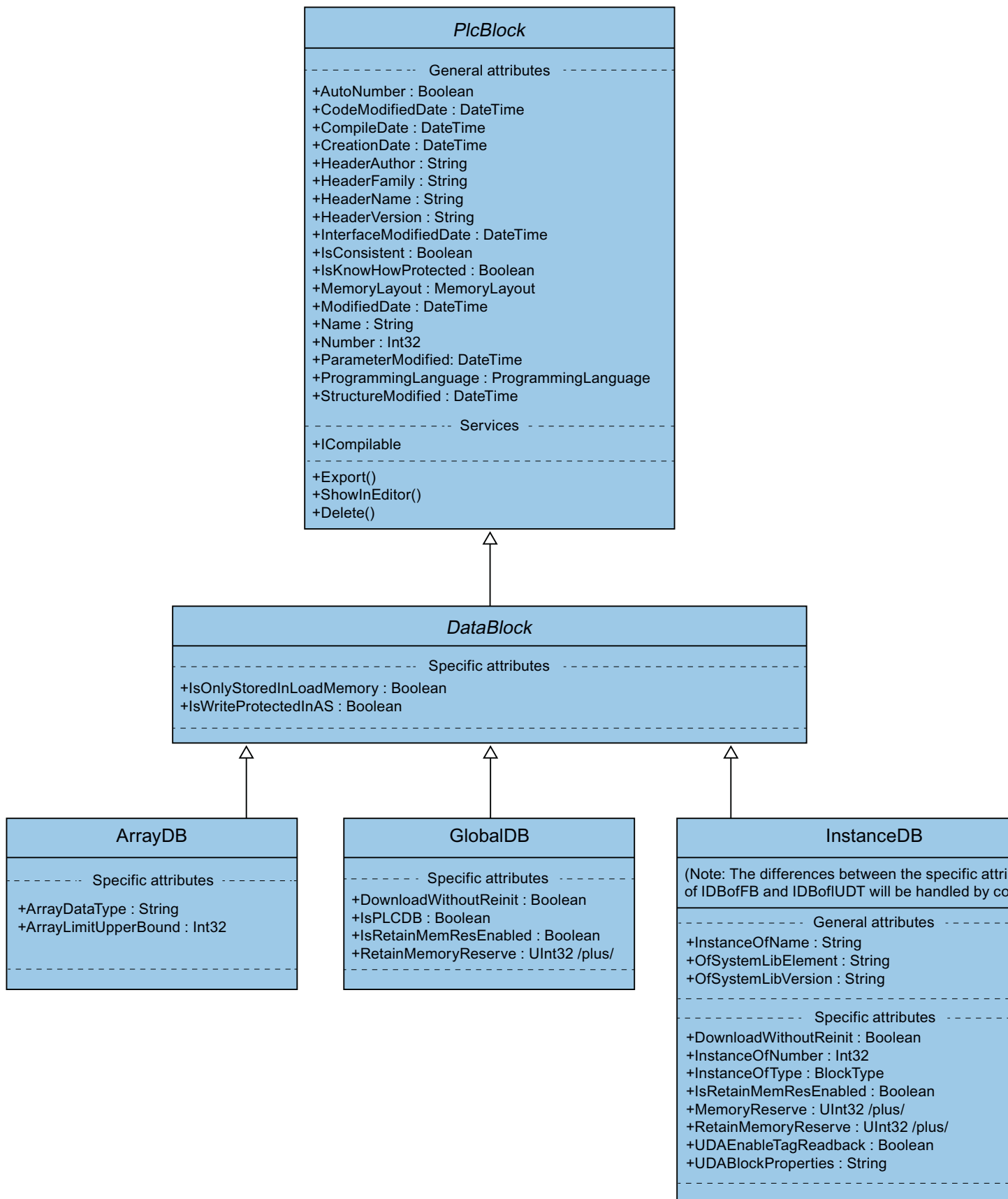
Représentation de blocs et types dans la TIA Portal Openness API

L'élément de modèle simplifié des blocs et de la structure est basé sur les attributs dans la TIA Portal Openness API. Les classes correspondantes mettent à disposition la fonction d'exportation ainsi que la fonction de compilation pour les blocs.

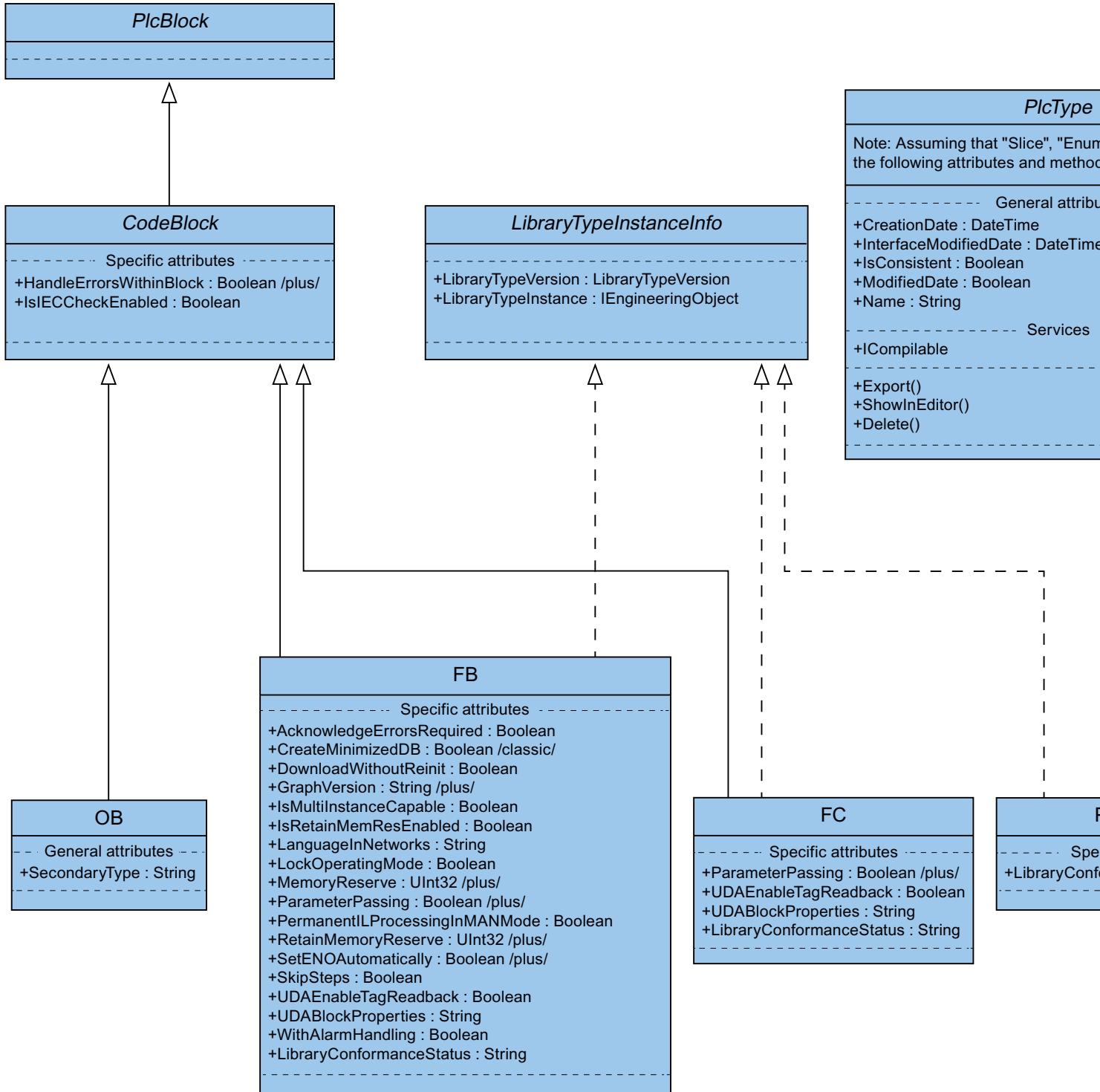
Diagrammes de classes

Toutes les classes qui ne sont pas directement instanciées sont définies de manière abstraite dans le modèle d'objet TIA Portal Openness.

Données



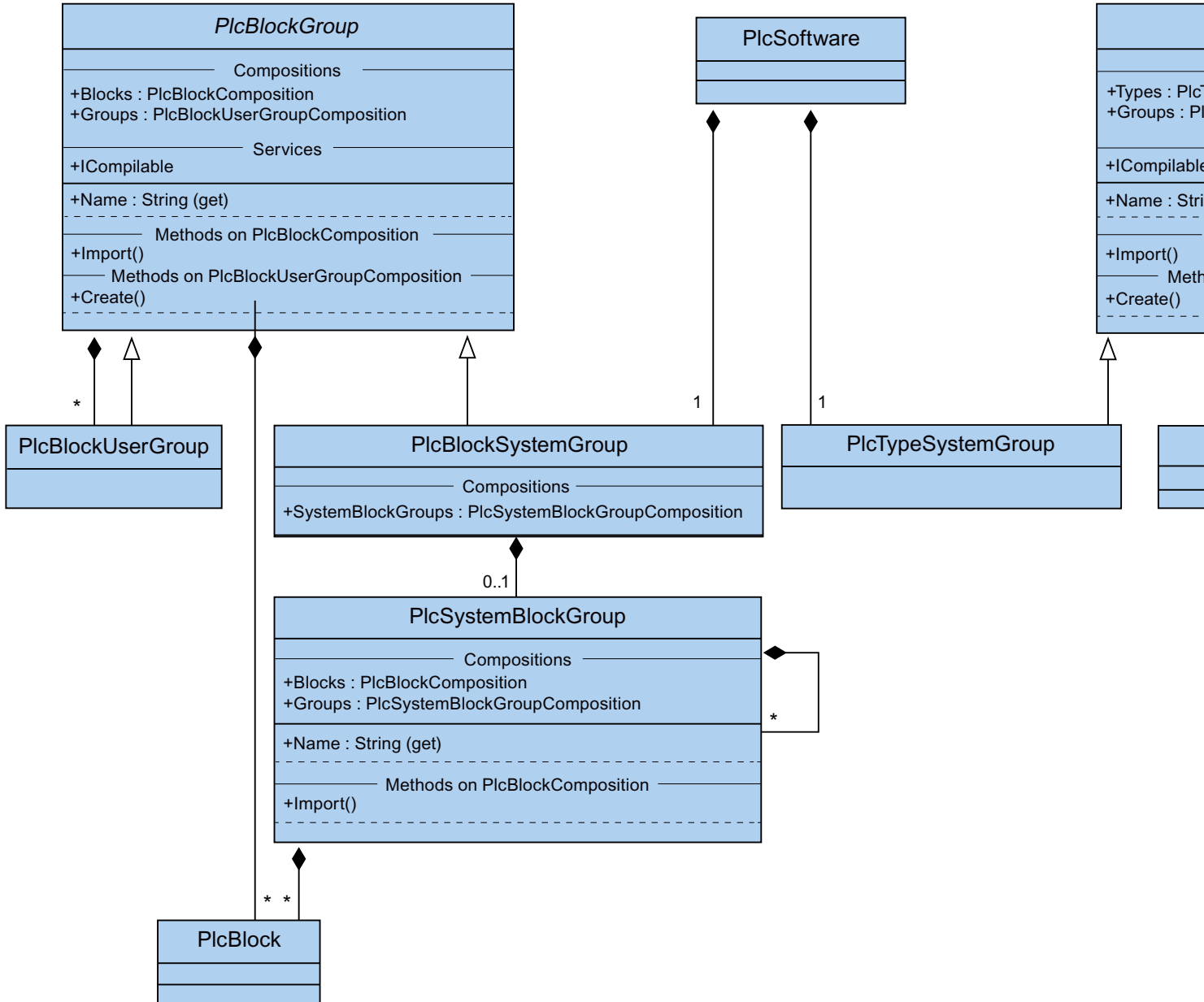
Code et type



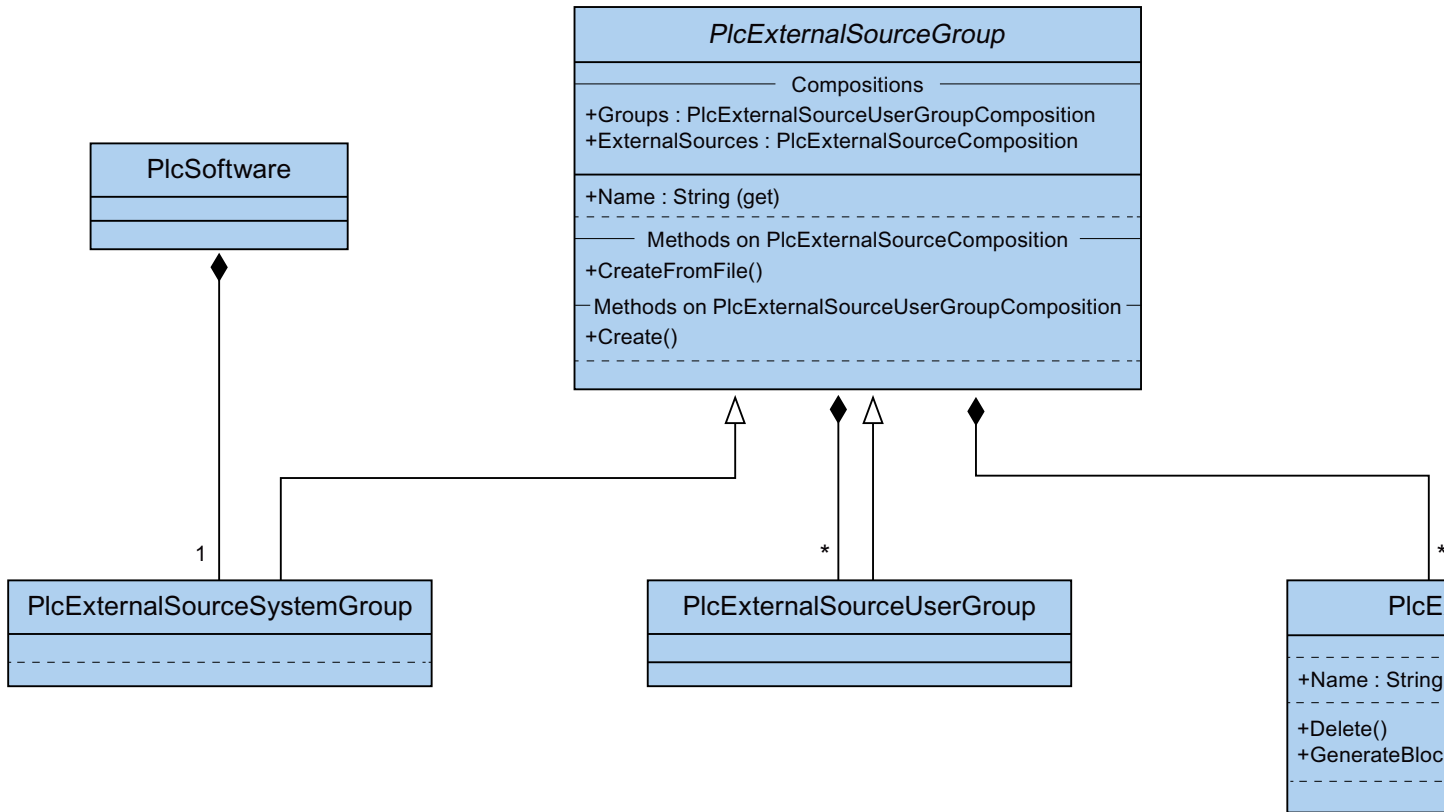
Représentation de groupes pour blocs et types dans la TIA Portal Openness API

Les deux groupes "PlcBlocks" du niveau supérieur ("Blocs de programme" dans l'interface utilisateur de TIA Portal) et "PlcTypes" ("Types de données API" dans l'interface utilisateur de TIA Portal) contiennent des blocs et des définitions de type. Ces groupes mettent la fonction d'importation et la fonction de compilation à la disposition des blocs. Étant donné que la plupart des méthodes qui s'appliquent aux fonctionnalités des groupes ne sont accessibles que via des bibliothèques, il existe une représentation "intégrée" ou "condensée" des bibliothèques et des méthodes correspondantes dans les classes "Host".

Blocs et types



Sources externes



Voir aussi

Modèle d'objet TIA Portal Openness (Page 51)

Hiérarchie des objets matériels du modèle d'objet (Page 64)

7.5 Hiérarchie des objets matériels du modèle d'objet

Relation entre les éléments visibles du portail TIA et les éléments structurés du modèle d'objet

Modèle matériel	Explication
Appareil (Device)	Objet conteneur pour une configuration centralisée ou décentralisée.
Élément d'appareil (Deviceltem)	Chaque objet élément d'appareil possède un objet conteneur. La relation logique est "Éléments".

Pour les objets éléments d'appareil, la relation conteneur-élément est similaire à la relation des modules.

Exemple : un appareil contient un ou plusieurs emplacements. Un emplacement contient des modules. Un module contient des sous-modules.

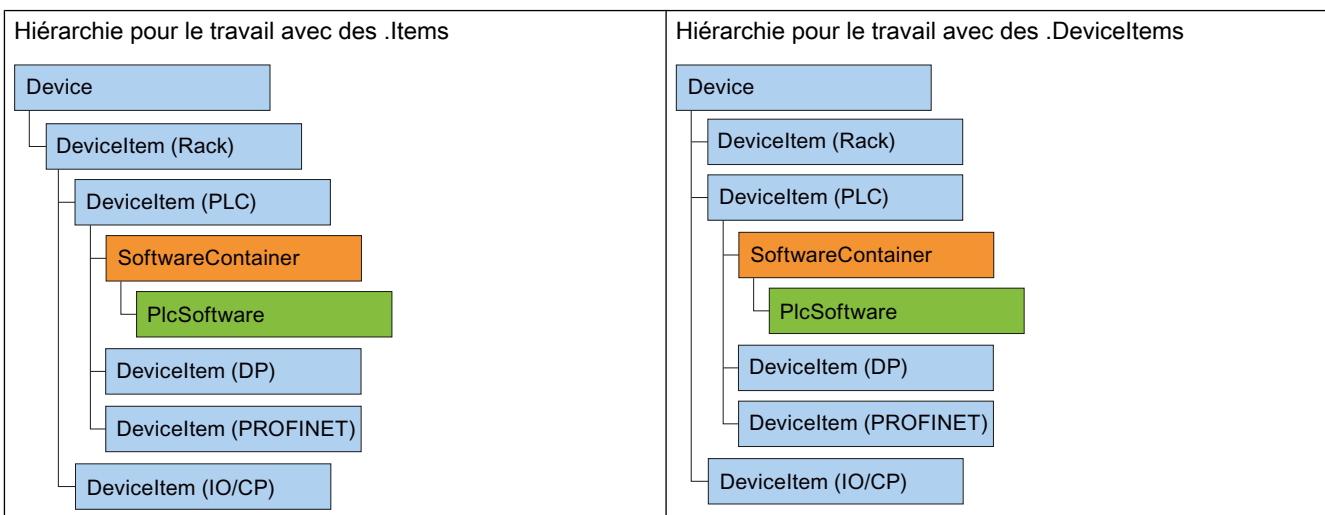
Il s'agit de la relation similaire à la représentation dans la vue de réseau et la vue d'appareil de TIA Portal. L'attribut "PositionNumber" d'un élément d'élément d'appareil est unique dans la zone du conteneur.

La relation enfant-parent entre les objets élément d'appareil est une relation logique pure dans le modèle d'objet. Un enfant ne peut exister sans ses parents.

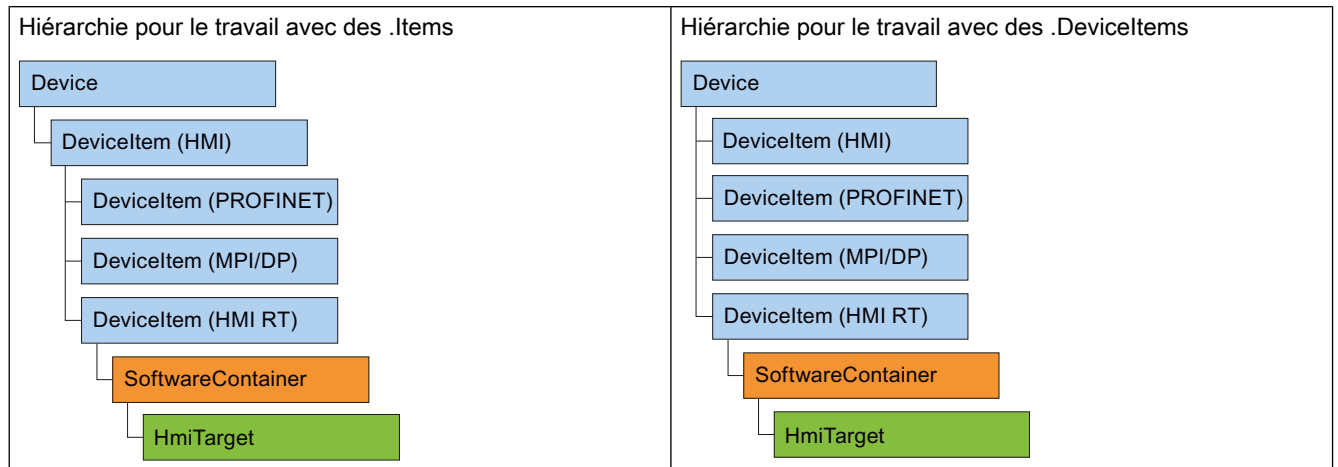
- Si un sous-module est structuré comme une partie d'un module (enfant), le sous-module ne peut pas être retiré sans le module.
- Si vous ajoutez au module un sous-module, puis vous le retirez du sous-module, cet enfant a les mêmes parents que le module.

Le diagramme ci-dessous indique la hiérarchie entre des appareils et des éléments d'appareils API et IHM.

Hiérarchie entre appareils API



Hiérarchie entre appareils IHM



Voir aussi

Modèle d'objet TIA Portal Openness (Page 51)

Blocs et types de modèle d'objet TIA Portal Openness (Page 56)

7.6 Informations sur les versions de TIA Portal Openness installées

Conditions

- TIA Portal Openness et TIA Portal sont installés

Utilisation

À partir de TIA Portal Openness V14, chaque version installée dispose d'une clé de Registre qui contient des informations sur la version. Cela permet de créer automatiquement le fichier de configuration d'application pour chaque version de TIA Portal Openness installée.

La clé de Registre se trouve dans le chemin suivant :

```
HKEY_LOCAL_MACHINE\Software\Siemens\Automation\Openness  
\14.0\PublicAPI
```

Remarque

Le numéro de version présent dans ce chemin est toujours identique au numéro de la version de TIA Portal actuelle installée. En cas de plusieurs installations parallèles, plusieurs jeux d'entrées correspondant à TIA Portal Openness sont présents dans le registre.

Il existe une clé unique par version de TIA Portal Openness. Les noms des versions sont identiques à ceux dans l'Assembly décrit, par ex. les entrées de registre pour TIA Portal Openness :

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\Openness\14.0\PublicAPI  
\14.0.1.0]"PublicKeyToken"="d29ec89bac048f84"  
"Siemens.Engineering"="C:\Program Files\Siemens\Automation\Portal V14\PublicAPI  
\V14\Siemens.Engineering.dll"  
"Siemens.Engineering.Hmi"="C:\Program Files\Siemens\Automation\Portal V14\PublicAPI  
\V14\Siemens.Engineering.Hmi.dll"  
"EngineeringVersion"="V14 SP1"  
"AssemblyVersion"="14.0.1.0"
```

Remarque

Si vous souhaitez créer un fichier de configuration d'application (Page 74), vous pouvez connaître le chemin d'accès à Siemens.Engineering.dll, Siemens.Engineering.Hmi.dll et au jeton de la clé publique grâce à la clé de Registre.

7.7 Exemple de programme

Exemple d'application : Créer l'accès de l'API dans une application

Le code du programme complet de l'exemple d'application est indiqué ci-après. Les étapes typiques de programmation sont décrites ci-après à l'aide de cet exemple.

Remarque

Un fichier de configuration d'exemple (Page 74) est requis pour l'exemple d'application.

7.7 Exemple de programme

```
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...");

                FileInfo projectPath = new FileInfo("C:\\Demo\\AnyCompanyProject.ap14"); //
edit the path according to your project
                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));

    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();
}
}
}
```

Procédure par étapes

1. Faire connaître le portail TIA dans l'environnement de développement

Créez dans votre environnement de développement un renvoi à tous les "fichiers dll" dans le répertoire "C:\Program Files\Siemens\Automation\PortalV..\PublicAPI\V..".

Cette procédure est illustrée ci-après pour le fichier "Siemens.Engineering.dll".

Le fichier "Siemens.Engineering.dll" se trouve dans le répertoire "C:\Program Files\Siemens\Automation\PortalV..\PublicAPI\V..". Créez dans votre environnement de développement un renvoi au fichier "Siemens.Engineering.dll".

Remarque

Affectez la valeur "False" au paramètre "CopyLocal" dans les attributs de référence.

2. Publier la plage de noms pour le TIA Portal

Insérez le code suivant :

```
using Siemens.Engineering;
```

3. Publier et démarrer TIA Portal

Pour publier et démarrer TIA Portal, insérez le code suivant :

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

4. Ouvrir un projet

Pour ouvrir un projet, vous pouvez par exemple utiliser le code suivant :

```
ProjectComposition projects = tiaPortal.Projects;  
Console.WriteLine("Opening Project...");  
FileInfo projectPath = new FileInfo("C:\\Demo\\AnyCompanyProject.ap14");  
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. Énumérer les appareils d'un projet

Ajoutez le code suivant pour énumérer tous les appareils du projet :

```
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. Enregistrer et fermer un projet

Insérez le code suivant puis enregistrez et fermez le projet :

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

7.8 Utilisation des exemples de code

Structure des sections de code

Chaque section de code de cette documentation est réalisée comme fonction sans valeur de retour avec une référence d'objet comme paramètre de transmission. Pour une meilleure lisibilité, on renoncera à supprimer des objets. L'accès aux objets de TIA Portal se fait par leur nom à l'aide de la méthode `Find`.

```
//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();
    }
}
```

Pour exécuter cette section de code, les éléments suivants sont requis :

- Un projet WinCC avec un appareil IHM contenant un groupe avec au moins une vue.
- Une fonction qui est instanciée par le pupitre opérateur.

Remarque

Si vous indiquez des chemins de répertoire, utilisez le chemin absolu, par ex. "C:/path/file.txt".

Les chemins de répertoire relatifs ne sont autorisés que pour les fichiers XML et l'importation/l'exportation, par ex. "file.txt" ou "C:/path01/.../path02/file.txt".

Exemple d'exécution de la section de code

Pour exécuter la section de code "DeleteScreenFromFolder" dans le cadre de l'exemple de programme "Hello TIA", utilisez l'exemple suivant :

```
//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();
    }
}
```

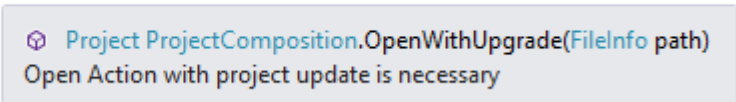
7.9 Fonctions générales

7.9.1 Prise en charge d'IntelliSense par TIA Portal Openness

Utilisation

Le support IntelliSense pour TIA Portal Openness vous offre de l'aide sur les attributs ou méthodes disponibles par le biais d'info-bulles. Il peut vous renseigner sur le nombre, les noms et les types de paramètres requis. Dans l'exemple suivant, le paramètre en gras dans la première ligne indique le paramètre requis suivant lors de l'entrée de la fonction.

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

A tooltip box with a light blue background and a small icon on the left. It contains the text: `Project ProjectComposition.OpenWithUpgrade(FileInfo path)` and below it, `Open Action with project update is necessary`.

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

Vous pouvez appeler manuellement des informations sur les paramètres de différentes manières : Cliquez sur "Edit IntelliSense/Parameter Info", à l'aide de la combinaison de touches <CTRL + MAJ + ESPACE> ou cliquez sur le bouton "Parameter Info" dans la barre d'outils de l'éditeur.

7.9.2 Etablissement d'une connexion au portail TIA

Introduction

Vous démarrez TIA Portal avec TIA Portal Openness ou établissez la liaison avec un TIA Portal en cours d'exécution. Si vous démarrez TIA Portal avec une application TIA Portal Openness, indiquez si TIA Portal doit être démarré avec ou sans interface utilisateur graphique. Si vous travaillez avec TIA Portal sans interface utilisateur, TIA Portal est uniquement lancé comme un processus du système d'exploitation. Si nécessaire, une application TIA Portal Openness vous permet de créer plusieurs instances de TIA Portal.

Remarque

Si vous utilisez l'application TIA Portal Openness pour accéder à l'interface de TIA Portal, vous ne pouvez pas utiliser d'éditeur IHM. Vous pouvez ouvrir l'éditeur "Appareils & réseaux" ou l'éditeur de programmation manuellement ou via TIA Portal Openness API.

Vous disposez des options suivantes pour démarrer TIA Portal avec une application TIA Portal Openness.

- Utilisez un fichier de configuration d'application (recommandé pour la plupart des applications).
- Utilisez la méthode "AssemblyResolve" (recommandé pour la copie, etc.).
- Copiez le fichier Siemens.Engineering.dll dans le répertoire de l'application TIA Portal Openness.

Remarque

Nous vous recommandons de charger Siemens.Engineering.dll à l'aide du fichier de configuration d'application. Cette méthode permet de tenir compte des noms forts et les modifications dommageables au Engineering.dll entraînent une erreur de chargement. Ce qui n'est pas détectable en cas d'utilisation de la méthode AssemblyResolve.

Démarrage de TIA Portal avec un fichier de configuration d'application

Créez dans le fichier de configuration d'application des renvois à toutes les bibliothèques de programmes requises. Répartissez le fichier de configuration d'application avec l'application TIA Portal Openness.

Enregistrez le fichier de configuration d'application "app.config" dans le même répertoire que l'application TIA Portal Openness et intégrez ce fichier dans votre application. Vérifiez dans chaque code si le chemin d'accès au fichier correspond au chemin d'installation de TIA Portal.

Vous pouvez utiliser l'extrait de code suivant pour le fichier de configuration d'application :

```
<?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 installation -->
        <codeBase version="14.0.1.0" href="FILE:///C:\Program Files\Siemens\Automation
\Portal V14\PublicAPI\V14 SP1\Siemens.Engineering.dll"/>
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="Siemens.Engineering.Hmi" culture="neutral"
publicKeyToken="d29ec89bac048f84"/>
        <!-- Edit the following path according to your installation -->
        <codeBase version="14.0.1.0" href="FILE:///C:\Program Files\Siemens\Automation
\Portal V14\PublicAPI\V14 SP1\Siemens.Engineering.Hmi.dll"/>
      </dependentAssembly>
    </assemblyBinding>
  </runtime>
</configuration>
```

Pour ouvrir une nouvelle instance de TIA Portal via le fichier de configuration d'application, utilisez le code de programme suivant :

```
//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
            }
        }
    }
}
```

Démarrage de TIA Portal avec la méthode "AssemblyResolve"

Créez le code de programme de TIA Portal Openness de telle sorte que l'enregistrement sur l'événement "AssemblyResolve" soit effectué le plus tôt possible. Encapsulez l'accès à TIA Portal dans un objet ou une méthode supplémentaire.

La prudence est de mise lors de la résolution d'Engineering Assembly par la méthode AssemblyResolve. Lorsque des types de l'Engineering Assembly sont utilisés avant l'exécution de l'Assembly Resolver, le programme se bloque. Cela est dû au fait que le compilateur Just-in-time (compilateur JIT) ne compile une méthode qu'au moment où il doit l'exécuter. Lorsque des types d'un Engineering Assembly sont utilisés par ex. dans "Main", le compilateur JIT essaie de compiler "Main" pendant l'exécution du programme. Ce qui échoue parce que le compilateur JIT ne sait pas où trouver l'Engineering Assembly. L'enregistrement de Assembly Resolver dans Main n'y change rien. La méthode doit être en cours d'exécution avant l'enregistrement de Assembly Resolver et compilée avant que celui-ci ne puisse être exécuté. La solution à ce problème consiste à placer la Business Logic, qui utilise les types provenant de l'Engineering Assembly, dans une méthode séparée. Et la méthode séparée utilise uniquement des types que le compilateur JIT comprend déjà. L'exemple suivant utilise une méthode qui fournit en retour "void", ne possède aucun paramètre et contient toutes les Business Logic. Le compilateur JIT peut maintenant compiler "Main" avec succès parce qu'il comprend tous les types présents dans Main. Assembly Resolver est déjà enregistré lorsque RunTiaPortal est appelé pendant l'exécution. Ainsi, le compilateur JIT sait où trouver l'Engineering Assembly lorsqu'il cherche les types de Business Logic.

Pour ouvrir une nouvelle instance de TIA Portal, utilisez le code de programme suivant.

```

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 installation
            string path = Path.Combine(@"C:\Program Files\Siemens\Automation\Portal
V14\PublicAPI\V14 SP1\", name);
            string fullPath = Path.GetFullPath(path);
            if (File.Exists(fullPath))
            {
                return Assembly.LoadFrom(fullPath);
            }
            return null;
        }
    }
}

```

Accéder à des instances actives de TIA Portal

Pour pouvoir établir une liaison à une instance active de TIA Portal avec une application TIA Portal Openness, énumérez d'abord les instances de TIA Portal. Vous pouvez vous connecter à plusieurs instances au cours d'une session Windows. L'instance active peut être TIA Portal avec ou sans interface utilisateur démarrée :

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

Si vous connaissez l'ID de processus de l'instance du TIA Portal, utilisez cet ID de processus pour accéder à l'objet. Le démarrage de TIA Portal nécessite un certain temps avant que vous ne puissiez le connecter à TIA Portal Openness.

Lors de l'établissement de la liaison à une instance active de TIA Portal, une invite de commande du pare-feu TIA Portal Openness s'affiche. Vous pouvez choisir parmi ces options pour la liaison :

- Autoriser la liaison de manière unique
 - Ne pas autoriser la liaison
 - Toujours autoriser les liaisons de cette application
- Pour plus d'informations, voir Pare-feu TIA Portal Openness (Page 79).

Remarque

Si l'invite de commande du registre est rejetée trois fois, le système déclenche une exception du type `EngineeringSecurityException`.

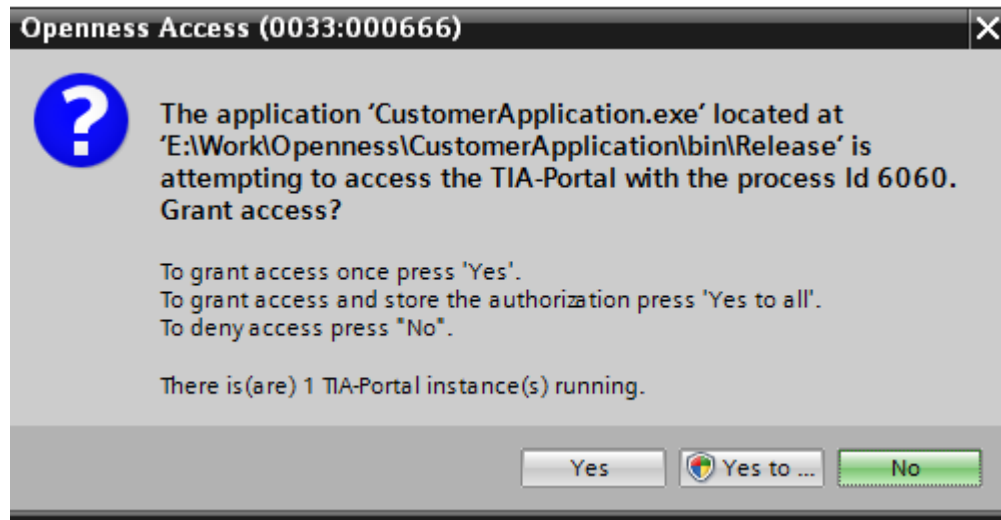
Une fois la liaison au processus établie, vous pouvez appeler des informations sur les instances de TIA Portal à l'aide de l'un des attributs suivants :

Attribut	Information
InstalledSoftware as <code>IList<TiaPortalProduct></code>	Fournit en retour des informations sur les produits installés.
Mode as <code>TiaPortalMode</code>	Fournit en retour le mode dans lequel TIA Portal a été démarré (<code>WithoutUserInterface/WithUserInterface</code>).
AttachedSessions as <code>IList<TiaPortalSession></code>	Fournit en retour une liste d'applications qui sont connectées à TIA Portal.
ProjectPath as <code>FileInfo</code>	Fournit en retour le nom de fichier du projet ouvert dans TIA Portal, y compris le dossier, par exemple. <code>"D:\WinCCProjects\ColorMixing\ColorMixing.ap14"</code> Si aucun projet n'est ouvert, une chaîne de caractères vide est fournie en retour.
ID as <code>int</code>	Fournit en retour l'ID de processus de l'instance TIA Portal.
Path as <code>FileInfo</code>	Fournit en retour le chemin d'accès au fichier exécutable de TIA Portal.

7.9.3 Pare-feu TIA Portal Openness

Invite de commande du pare-feu TIA Portal Openness

Si vous essayez d'établir une liaison via TIA Portal Openness à une instance de TIA Portal en cours d'exécution, TIA Portal vous invite à accepter ou à rejeter la liaison, comme représenté sur la copie d'écran suivante.



Autoriser la liaison à TIA Portal de manière unique

Si vous souhaitez autoriser de manière unique la liaison de votre application TIA Portal Openness à TIA Portal, cliquez sur "Yes" à l'invite de commande. Lorsque votre application TIA Portal Openness essaiera d'établir une liaison à TIA Portal la prochaine fois, l'invite apparaîtra de nouveau.

Ajouter une entrée Whitelist par l'accès à TIA Portal

Pour créer une entrée Whitelist pour votre application TIA Portal Openness, procédez comme suit :

1. Lorsque vous cliquez sur "Yes to all" à l'invite, une boîte de dialogue s'ouvre pour le contrôle de compte utilisateur.
2. Dans cette boîte de dialogue, cliquez sur "Yes" pour ajouter votre application pour la Whitelist dans la base de données d'enregistrement Windows et pour relier l'application à TIA Portal.

Ajouter une entrée Whitelist sans TIA Portal

Si vous voulez créer une entrée Whitelist sans utiliser TIA Portal, vous pouvez générer un fichier de Registre comme suit :

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\Openness\14.0\Whitelist
\CustomerApplication.exe]
[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\Openness\14.0\Whitelist
\CustomerApplication.exe\Entry]
"Path"="E:\\Work\\Openness\\CustomerApplication\\bin\\Release\\CustomerApplication.exe"
"DateModified"="2014/06/10 15:09:44.406"
"FileHash"="0rXRKUCNzMWHOMFrT52OwXzqJef10ran4UykTeBraaY="
```

L'exemple suivant montre comment calculer la valeur Filehash et la date de dernière modification :

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

7.9.4 Gestionnaire d'événements

Gestionnaire d'événements dans l'application TIA Portal Openness

Une instance de TIA Portal propose les événements suivants, auxquels vous pouvez réagir avec un gestionnaire d'événements dans une application TIA Portal Openness. Vous pouvez accéder aux attributs de notifications et définir les réactions en conséquence.

Événement	Réponse
Disposed	Cet événement vous permet de réagir avec une application TIA Portal Openness à la fermeture de TIA Portal.
Notification	Cet événement vous permet de réagir avec une application TIA Portal Openness aux notifications de TIA Portal. Les notifications requièrent juste une confirmation, telle que "OK".
Confirmation	Cet événement vous permet de réagir avec une application TIA Portal Openness aux confirmations de TIA Portal. Les confirmations requièrent toujours une décision, telle que "Voulez-vous enregistrer le projet ?".

Code du programme

Pour enregistrer un gestionnaire d'événements dans une application TIA Portal Openness, modifiez le code de programme suivant :

```
//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)
{
    ....
}
```

Attributs des notifications du TIA Portal

Les notifications du TIA Portal possèdent les attributs suivants :

Attribut	Description
Caption	Fournit en retour le nom de la confirmation.
DetailText	Fournit en retour le texte détaillé de la confirmation.
Icon	Fournit en retour l'icône de la confirmation.
IsHandled	Fournit en retour la confirmation ou indique si le système attend la confirmation.
MessageID	Fournit en retour l'ID univoque au sein du service.
ServiceID	Fournit en retour l'ID du service ayant déclenché la confirmation.
Text	Fournit en retour le texte de la confirmation.

Attributs des confirmations

Les confirmations possèdent les attributs suivants :

Attribut	Description
Caption	Fournit en retour le nom de la confirmation.
Choices	Donne les possibilités pour acquitter la confirmation.
DetailText	Fournit en retour le texte détaillé de la confirmation.
Icon	Fournit en retour l'icône de la confirmation.
IsHandled	Fournit en retour la confirmation ou indique si le système attend la confirmation.
MessageID	Fournit en retour l'ID univoque au sein du service.
Result	Fournit ou indique le résultat de l'acquiescement.
ServiceID	Fournit en retour l'ID du service ayant déclenché la confirmation.
Text	Fournit en retour le texte de la confirmation.

Voir aussi

Confirmer les boîtes de dialogue comportant des alarmes système par commande du programme (Page 82)

7.9.5 Confirmer les boîtes de dialogue comportant des alarmes système par commande du programme

Condition requise

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Les gestionnaires d'événements sont enregistrés.
Voir Etablissement d'une connexion au portail TIA (Page 74)

Utilisation

Si vous commandez le portail TIA par le biais de l'interface utilisateur, des boîtes de dialogue comportant des événements système s'affichent pour certaines séquences du programme. À l'aide de ces événements système, vous décidez comment continuer.

Si vous accédez à TIA Portal avec une application TIA Portal Openness, ces événements système doivent être acquittés via les événements ".NET" correspondants.

Les confirmations autorisées figurent dans la liste `Choices` :

- Abort
- Cancel
- Ignore

- No
- NoToAll
- None
- OK
- Retry
- Yes
- YesToAll

La valeur de `ConfirmationEventArgs.Result` doit être l'une des entrées susmentionnées. Faute de quoi, une exception est lancée.

Code du programme

Pour réagir à un événement de confirmation, modifiez le code de programme suivant :

```
private void TiaPortalOnConfirmation(object sender, ConfirmationEventArgs e)
{
    int serviceId = e.ServiceId;
    int messageId = e.MessageId;
    if (serviceId == MyExpectedServiceId)
    {
        if (messageId == MyExpectedMessageId && ((e.Choices & ConfirmationChoices.Yes) == ConfirmationChoices.Yes))
        {
            e.Result = ConfirmationResult.Yes;
            e.IsHandled = true;
        }
    }
}
```

Pour informer le concepteur des actions exécutées avec une application TIA Portal Openness, modifiez le code de programme suivant :

```
//Handles notifications
using (TiaPortal tiaPortal = new TiaPortal())
{
    tiaPortal.Notification += Notification;
    try
    {
        //perform actions that will result in a notification event
    }
    finally
    {
        tiaPortal.Notification -= Notification;
    }
}
```

```
private void TiaPortalOnNotification(object sender, NotificationEventArgs e)
{
    if (e.MessageId == MyExpectedMessageId && e.ServiceId == MyExpectedServiceId)
    {
        e.IsHandled = true;
    }
}
```

7.9.6 Mettre fin à la connexion au portail TIA

Introduction

Si vous avez démarré l'instance du TIA Portal sans interface utilisateur et que votre application est le seul client TIA Portal Openness relié à TIA Portal, vous pouvez fermer cette instance du TIA Portal via l'application TIA Portal Openness. Sinon, déconnectez l'application TIA Portal Openness de l'instance du TIA Portal.

Déconnectez ou fermez l'instance active de TIA Portal à l'aide de la méthode `IDisposable.Dispose()`.

Vous pouvez utiliser la méthode `IDisposable.Dispose()` comme suit :

- avec un using-Statement.
- Entourez la description de l'objet avec un bloc try-finally et appelez la méthode `IDisposable.Dispose()` au sein du bloc finally.

Si vous quittez l'instance active du portail TIA, vous ne pouvez plus accéder au portail TIA.

Remarque

Si un concepteur ferme l'instance du TIA Portal en dépit d'un accès en cours d'une application TIA Portal Openness, une exception de la classe "NonRecoverableException" est déclenchée lors du prochain accès dans l'application TIA Portal Openness. Vous pouvez vous abonner à un événement `Disposed` afin de recevoir un appel lors de la fermeture de TIA Portal.

Code du programme

Pour couper la connexion au portail TIA ou y mettre fin, utilisez le code de programme suivant :

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

Voir aussi

Gestionnaire d'événements (Page 80)

7.9.7 Interfaces de diagnostic dans TIA Portal**Utilisation**

Vous pouvez appeler certaines informations de diagnostic d'instances de TIA Portal en cours d'exécution avec une méthode statique. L'interface de diagnostic est réalisée dans l'objet TiaPortalProcess que vous pouvez appeler pour chaque instance TIA Portal en cours d'exécution.

L'interface de diagnostic n'est pas bloquée. Vous pouvez donc accéder à l'objet TiaPortalProcess ainsi qu'à ses membres et ce, que TIA Portal soit occupé ou non. L'interface de diagnostic comprend les membres suivants :

Classe TiaPortalProcess

Membre	Type	Fonction
AcquisitionTime	DateTime	Date et heure auxquelles l'objet TiaPortalProcess a été saisi. Comme l'objet TiaPortalProcess représente un instantané tout à fait statique de l'état de TIA Portal à un moment donné, les informations qu'il contient peuvent être obsolètes.
Attach	TiaPortal	Est relié à TiaPortalProcess donné et fournit en retour une instance de TIA Portal.
AttachedSessions	IList<TiaPortalSession>	Bibliothèque de toutes les autres sessions actuellement attachées au même TIA Portal. Cette bibliothèque peut être vide. Chaque session est représentée par un objet TiaPortalSession décrit comme suit.

Membre	Type	Fonction
Attaching	EventHandler<AttachingEventArgs>	Cet événement permet à l'application d'autoriser des tentatives de lien à TIA Portal. Lorsqu'une autre application essaie de s'attacher à TIA Portal, les membres de cet événement en sont informés. Les membres ont alors 10 secondes pour autoriser l'opération. Si un membre ignore cet événement ou n'y réagit pas à temps, cela est considéré comme un refus et l'opération n'est pas autorisée à l'application demandeuse. Les applications bloquées, et donc incapables de réagir à cet événement, ne peuvent pas refuser à une autre application l'autorisation de s'attacher.
Dispose	void	Ferme l'instance de TIA Portal correspondante.
Id	int	ID de processus de TIA Portal.
InstalledSoftware	IList<TiaPortalProduct>	Bibliothèque de tous les produits actuels installés comme partie de TIA Portal. Chaque produit est représenté par un objet TiaPortalProduct décrit comme suit.
Mode	TiaPortalMode	Fournit en retour le mode dans lequel TIA Portal a été démarré. Valeurs actuelles : WithUserInterface et WithoutUserInterface.
Path	FileInfo	Chemin d'accès au fichier exécutable de TIA Portal.
ProjectPath	FileInfo	Chemin d'accès au projet actuel ouvert dans TIA Portal. Si aucun projet n'est ouvert, cet attribut a la valeur zéro.

Classe TiaPortalSession

Membre	Type	Fonction
AccessLevel	TiaPortalAccessLevel	Niveau d'accès de la session. Est représenté sous forme d'une énumération de mementos et plusieurs niveaux d'accès sont possibles. La section qui suit propose une description détaillée de TiaPortalAccessLevel.
AttachTime	DateTime	Date et heure auxquelles la liaison à TIA Portal a été établie.
Id	int	L'ID de la session actuelle.

Membre	Type	Fonction
IsActive	bool	Fournit "vrai" en retour lorsque le traitement d'un appel provenant de cette session est en cours d'exécution sur TIA Portal.
Dispose	void	Met fin à la connexion entre le processus et TIA Portal. Cette méthode ne force pas la fin du processus, comme ce serait le cas avec System.Diagnostics.Process.Kill. L'application dont la connexion est terminée reçoit malgré tout un événement Disposed. Toutefois, aucune indication n'est donnée sur ce qui a provoqué la fin de la connexion.
ProcessId	int	ID du processus attaché.
ProcessPath	FileInfo	Chemin d'accès au fichier exécutable du processus attaché.
TrustAuthority	TiaPortalTrustAuthority	Indique si la session actuelle a été démarrée par un processus signé et s'il s'agit d'un certificat TIA Portal Openness ou non. TrustAuthority est une énumération de mémentos décrite comme suit.
UtilizationTime	TimeSpan	Période pendant laquelle le processus a été actif dans TIA Portal. En combinaison avec l'attribut AttachTime, ceci peut servir à déterminer des pourcentages de durée de vie ou d'autres informations similaires.
Version	string	Version de Siemens.Engineering.dll attaché à la session.

Enum TiaPortalAccessLevel

Valeur d'énumération	Fonction
None	Cette valeur n'est pas valide. La valeur est indiquée parce que TiaPortalAccessLevel est un memento de type Enum et qu'à ce titre, une "valeur zéro" correspondante est requise pour montrer qu'aucun memento n'est mis à 1. Mais elle n'apparaît jamais dans la pratique, car aucune session ne peut être démarrée sans accès.
Published	La session a accès à la fonctionnalité publiée.
Modify	La session a accès en modification.

Enum TiaPortalTrustAuthority

Valeur d'énumération	Fonction
None	Le module principal du processus attaché n'est pas signé à l'aide d'un certificat.
Signed	Le module principal est signé à l'aide d'un certificat, mais il ne s'agit pas d'un certificat TIA Portal Openness.
Certified	Le module principal est signé à l'aide d'un certificat TIA Portal Openness.
CertifiedWithExpiration	Le module principal est signé à l'aide d'un certificat TIA Portal Openness qui perd sa validité lorsque sa durée de vie est écoulée.

Classe TiaPortalProduct

Membre	Type	Fonction
Name	string	Nom du produit (par ex. STEP 7 Professional).
Options	IList<TiaPortalProduct>	Une bibliothèque de tous les progiciels qui font partie du TIA Portal relié, représentés comme objets TiaPortalProduct. Lorsqu'un progiciel contient, à son tour, des progiciels, le niveau d'imbrication peut être plus important.
Version	string	Trame de version du produit.

L'exemple de section de code suivant montre comment utiliser l'interface de diagnostic pour interroger des informations et comment ensuite utiliser ces informations dans votre application.

```

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);
    }
}

```

Information de sécurité

Du fait que l'utilisation de l'interface de diagnostic ne nécessite aucune liaison à TIA Portal, il est possible d'écrire un service Windows qui se sert de l'événement latéral pour vérifier toute application qui essaie de s'attacher à TIA Portal. Ainsi, il est possible par ex. de définir que seules les applications commençant par le nom de votre entreprise sont autorisées à s'attacher. Une autre option pourrait consister à autoriser systématiquement l'accès mais à écrire les informations sur les processus latéraux dans un journal. Le code de programme suivant donne un exemple de gestionnaire d'événements pour vérifier des liaisons entrantes :

```
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();
    }
}
```

7.9.8 Exclusive access

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Pour configurer un processus exclusif d'accès à un processus de TIA Portal attaché, la classe "TIA Portal" propose la méthode "ExclusiveAccess(String text)". L'utilisation d'un accès exclusif est fortement recommandée même si ce dernier n'est pas obligatoire.

Utilisez "ExclusiveAccess" dans une instruction "using" pour vous assurer que l'accès est correctement terminé lorsque des exceptions se produisent ou l'application est arrêtée.

Remarque

Toute tentative visant à créer un deuxième accès exclusif dans l'étendue d'un accès exclusif ouvert, entraîne le déclenchement d'une exception qui peut être restaurée.

Modifiez l'exemple suivant pour avoir "ExclusiveAccess" à une instance :

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

Après la saisie d'une instance "ExclusiveAccess" pour un processus de TIA Portal donné, une boîte de dialogue s'ouvre. Cette boîte de dialogue affiche le message prévu lors de l'instanciation. En plus, les informations suivantes sur l'application Client sont également affichées :

- le titre d'Assembly des données manifestes, s'il est disponible, ou le nom du processus
- l'ID du processus
- le SID

Remarque

Plusieurs sessions peuvent être actives pour une application Client TIA Portal Openness donnée parce qu'il existe plusieurs instances de TIA Portal, même si celles-ci sont attribuées respectivement au même processus TIA Portal.

L'application Client peut également actualiser le contenu affiché dans la boîte de dialogue d'accès exclusif en remplaçant les valeurs de l'attribut "Texte" par de nouvelles valeurs. Pour obtenir ce comportement, modifiez le code de programme suivant :

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

Vous pouvez exiger l'annulation de l'accès exclusif à l'aide du bouton "Cancel"/"Annuler". Pour obtenir ce comportement, modifiez le code de programme suivant :

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

7.9.9 Traitement des transactions

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Opération

Une rémanence (projet, bibliothèque, etc.) ouverte au sein d'un processus TIA Portal correspondant peut être modifiée par une application Client TIA Portal Openness. Vous pouvez obtenir cette modification avec une opération unique ou une série d'opérations. Pendant l'action, il est utile de regrouper ces opérations dans une seule pile Undo afin d'obtenir un déroulement plus logique. En outre, regrouper des opérations dans une pile Undo unique présente également des avantages en matière de performance. Pour aider à cela, la classe "ExclusiveAccess" propose la méthode "Transaction(ITransactionSupport persistence, string undoDescription)". L'appel depuis cette méthode entraîne l'instanciation d'un nouvel objet de type "Transaction" qui peut être arrêté. Vous devez fournir la description du contenu de la transaction (l'attribut Texte ne doit pas être nul ou vide). Tant que cette instance n'a pas été terminée, toutes les opérations des applications Client sont regroupées dans une pile Undo unique au sein du processus TIA Portal correspondant.

Pour saisir une instance de type "Transaction", modifiez le code de programme suivant :

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

Remarque

Utilisez une instruction "using" pour instancier une "Transaction". Cela vous permet d'assurer que la transaction soit terminée correctement, même si des exceptions se produisent, et d'annuler ainsi la transaction.

Application cohérente ou annulation

Une "Transaction" dans une application Client vous permet d'assurer qu'il existe un chemin prévisible pour rendre effectif ou annuler un jeu de modifications. Votre application Client doit décider si les modifications apportées à une rémanence doivent devenir effectives ou pas. Pour cela, votre application doit demander que les modifications dans l'étendue d'une transaction ouvertes deviennent effectives lorsque la transaction est terminée par appel de la méthode "Transaction.CommitOnDispose()". Si cette méthode n'est jamais appelée dans le déroulement du code, les modifications apportées dans l'étendue de la transaction ouverte sont automatiquement annulées à la fin de la transaction.

Si une exception se produit après cette requête, toutes les modifications apportées dans l'étendue d'une transaction ouverte sont malgré tout annulées à la fin de la transaction.

Pour créer une pile Undo unique dans un processus TIA Portal attaché avec deux modifications "Create", modifiez le code de programme suivant :

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

Restrictions

Les actions suivantes ne sont pas autorisées dans une transaction. L'appel de ces actions entraîne une exception récupérables :

- Compile
- Go Online
- Go Offline

- ProjectText Import
- ProjectText Export
- Open Global Library
- Close Global Library
- Project Create
- Project Open
- Project OpenWithUpgrade
- Project Save
- Project Close

Comportement Undo

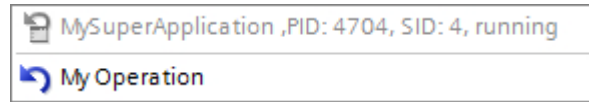
Les actions exécutées par une application Client TIA Portal Openness peuvent entraîner des piles Undo au sein du processus TIA Portal attaché. Chacune de ces entrées Undo est regroupée sous une entrée locale. L'entrée locale est composée des informations suivantes provenant de l'application Client :

- le titre d'Assembly des données manifestes, s'il est disponible, ou le nom du processus
- l'ID du processus
- le SID ou
- un message indiquant que le processus Client est en cours d'exécution

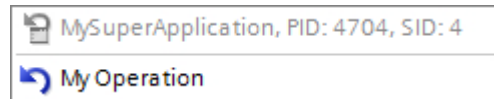
Ces entrées appartiennent à un des deux types suivants :

1. Les opérations, qui sont regroupées dans une transaction Undo comme résultats de l'utilisation d'une "Transaction", possèdent la description mise à disposition par l'application Client lorsque la "Transaction" a été instanciée.

- Entrée Undo pour une application Client en cours d'exécution :

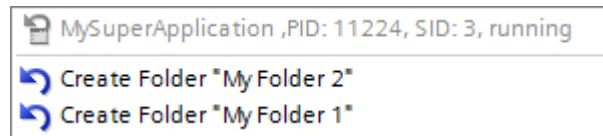


- Entrée Undo pour une application Client arrêtée :

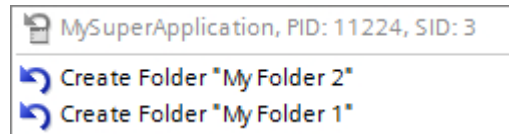


2. Pour les opérations qui sont exécutées individuellement, il existe des entrées Undo séparées pour la description de l'opération telle que définie dans la commande de métadonnées correspondante.

- Entrée Undo pour une application Client en cours d'exécution :



- Entrée Undo pour une application Client arrêtée :



7.9.10 Créer un objet DirectoryInfo/FileInfo

Utilisation

Les instances des classes `DirectoryInfo` et `FileInfo` doivent contenir un chemin absolu. Dans le cas contraire, les méthodes qui utilisent les objets `DirectoryInfo` ou `FileInfo` provoquent une exception.

Code de programme

Pour créer un objet `DirectoryInfo` ou `FileInfo`, modifiez le code de programme suivant :

```

..
//Create a DirectoryInfo object
string directoryPath = @"D:\Test\Project 1";
DirectoryInfo directoryInfo = new DirectoryInfo(directoryPath);

//Create a FileInfo object
string fileName = @"D:\Test\Project 1\Project 1.ap14";
FileInfo fileInfo = new FileInfo(fileName);
...

```

7.9.11 Prise en charge de l'autodescription pour attributs, navigateurs, actions et services

Utilisation

Dans TIA Portal Openness, chaque `IEngineeringServiceProvider` de la TIA Portal Openness API décrit ses capacités pour de potentiels appels.

Prise en charge de l'autodescription pour `IEngineeringObject`

Nom de la méthode	Valeurs retournées
<code>GetCompositionInfos</code>	Fournit en retour une bibliothèque d'objets du type <code>EngineeringCompositionInfo</code> , qui décrivent les différentes compositions de ces objets. La section qui suit propose une description de <code>EngineeringCompositionInfo</code> .
<code>GetAttributeInfos</code>	Fournit en retour une bibliothèque d'objets du type <code>EngineeringAttributeInfo</code> , qui décrivent les différents attributs de ces objets. La section qui suit propose une description de <code>EngineeringAttributeInfo</code> .
<code>GetInvocationInfos</code>	Fournit en retour une bibliothèque d'objets du type <code>EngineeringInvocationInfo</code> , qui décrivent les différentes actions de ces objets. La section qui suit propose une description de <code>EngineeringInvocationInfo</code> .

Prise en charge de l'autodescription pour `IEngineeringServiceProvider`

Nom de la méthode	Valeurs retournées
<code>GetServiceInfos</code>	Fournit en retour une bibliothèque d'objets du type <code>EngineeringServiceInfo</code> , qui décrivent les différents services de ces objets. La section qui suit propose une description de <code>EngineeringServiceInfo</code> .

Classe EngineeringCompositionInfo

Nom d'attribut	Valeurs retournées
Name	Nom de la composition.

Classe EngineeringAttributeInfo

Nom d'attribut	Valeurs retournées
AccessMode	Niveau d'accès pris en charge par l'attribut. La section qui suit propose une description détaillée de cet attribut.
Name	Nom de l'attribut.

Classe EngineeringInvocationInfo

Nom d'attribut	Valeurs retournées
Name	Nom de l'action.
ParameterInfos	Une bibliothèque d'objets du type EngineeringInvocationParameterInfo, qui décrivent les paramètres requis éventuellement pour l'action. La section qui suit propose une description de EngineeringInvocationParameterInfo.

Classe EngineeringServiceInfo

Nom d'attribut	Valeurs retournées
Type	Type de service comme objet System.Type.

Enum AccessMode

Valeur d'énumération	Valeurs retournées
None	Option invalide.
Read	L'attribut peut être lu.
Write	L'attribut peut être écrit.

Classe EngineeringInvocationParameterInfo

Nom d'attribut	Valeurs retournées
Name	Nom du paramètre.
Type	Type du paramètre comme objet System.Type.

Code de programme

AccessMode est une énumération de mementos dont les valeurs peuvent être combinées comme dans le code de programme suivant :

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

Pour rechercher tous les attributs d'un IEngineeringObject et apporter des modifications au mode d'accès à ces derniers, modifiez le code de programme suivant :

```
...
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;
    }
}
...
```

7.10 Fonctions des projets et données de projet

7.10.1 Ouvrir un projet

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Le projet à ouvrir n'est ouvert dans aucune autre instance de TIA Portal.

Remarque

Annulation de la mise à niveau d'un projet

Si vous annulez la mise à niveau d'un projet à V14 SP1 après avoir relié ce projet à TIA Portal Openness, des conflits se produisent.

Utilisation

Utilisez la méthode `Projects.Open` pour ouvrir un projet. Dans la méthode `Projects.Open`, entrez un chemin pour le projet de votre choix.

La méthode `Projects.Open` ne peut accéder qu'aux projets qui ont été créés ou mis au niveau à la version la plus récente de TIA Portal . Si vous accédez à un projet d'une version précédente avec la méthode `Projects.Open` , vous obtiendrez une exception en retour. Utilisez la méthode `OpenWithUpgrade` pour ouvrir les projets qui ont été créés avec une version précédente de TIA Portal.

Remarque

Pas d'accès aux projets en lecture seule

TIA Portal Openness ne peut accéder qu'aux projets autorisant la lecture et l'écriture.

Code de programme

Pour ouvrir un projet, modifiez le code de programme suivant :

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

Ouverture d'un projet protégé par UMAC

L'utilisateur peut également ouvrir un projet protégé par UMAC. La surcharge de la fonction Open nécessite un paramètre supplémentaire de type UmacDelegate. Ce paramètre supplémentaire permet à l'appelant d'indiquer un gestionnaire à utiliser pendant l'authentification UMAC. Le nouveau UmacDelegate est mis en œuvre avec une méthode contenant un paramètre de type UmacCredentials. UmacCredentials a deux propriétés, 'Name' de type String et 'Type' de type UmacUserType, ainsi qu'une méthode SetPassword avec un paramètre de type SecureString. L'utilisation de UmacUserType.Project permet d'indiquer un volume de projet UMAC, alors que l'utilisation de UmacUserType.Global indique un volume d'application UMAC (c'est-à-dire commandé par un serveur UMAC).

Code de programme

```
...
    Siemens.Engineering.Project project = tiaPortal.Projects.Open(new FileInfo(@"C:\Some\Path
\Here\Project.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);
}
...
}
```

Ouvrir un projet qui a été créé avec une version précédente

Utilisez la méthode `OpenWithUpgrade` pour ouvrir un projet qui a été créé avec la version précédente de TIA Portal. La méthode crée un nouveau projet actualisé et l'ouvre.

Si vous accédez à un projet créé avec une version plus ancienne, vous obtiendrez une exception en retour.

Remarque

Si vous accédez à un projet créé avec la version la plus récente, le projet s'ouvre.

Code de programme

Modifiez le code de programme suivant pour ouvrir un projet avec la méthode `OpenWithUpgrade` :

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

Code de programme pour un projet protégé avec UMAC

L'utilisateur peut également ouvrir un projet protégé par UMAC qui a été créé avec une version précédente de TIA Portal. Une fonction de surcharge `OpenWithUpgrade` nécessite un paramètre supplémentaire de type `UmacDelegate`.

```
...
Siemens.Engineering.Project project = tiaPortal.Projects.OpenWithUpgrade(new
FileInfo(@"C:\Some\Path\Here\Project.apXX"), MyUmacDelegate);
...
```

7.10.2 Créer un projet

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal. Voir AUTOHOTSPOT

Utilisation

La TIA Portal Openness API permet de créer des projets

- par appel de la méthode `Create` sur `ProjectComposition`
- par appel de la méthode `Create` sur `IEngineeringComposition`

ProjectComposition.Create

Modifiez le code de programme suivant :

```
TiaPortal tiaPortal = ...;
ProjectComposition projectComposition = tiaPortal.Projects;
DirectoryInfo targetDirectory = new DirectoryInfo(@"D:\TiaProjects");

// Create a project with name MyProject
Project project = projectComposition.Create(targetDirectory, "MyProject");
```

Dans l'exemple utilisé,

- un dossier "D:\TiaProjects\MyProject" est généré.
- un fichier de projet "D:\TiaProjects\MyProject\MyProject.aPXX" est créé.

Remarque

Paramètre targetDirectory

Le paramètre targetDirectory peut également représenter un chemin UNC (Universal Naming Convention). Un projet peut donc également être créé sur un lecteur validé dans le réseau.

IEngineeringComposition.Create

Modifiez le code de programme suivant :

```
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);
```

Dans l'exemple utilisé,

- un dossier "D:\TiaProjects\MyProject" est généré.
- un fichier de projet "D:\TiaProjects\MyProject\MyProject.aPXX" avec les attributs de projet Auteur = "Bob" et Commentaire = "This project was created with openness" est créé.

Paramètres pour la création d'un projet avec des attributs de projet optionnels

Paramètre	Type de données	Obligatoire	Description
Author	String	non	Auteur d'un projet.
Comment	String	non	Commentaire relatif au projet.
Name	String	oui	Nom d'un projet.
TargetDirectory	DirectoryInfo	oui	Répertoire contenant le dossier de projet créé.

7.10.3 Accéder à des paramètres généraux de TIA Portal

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Vous pouvez accéder à des paramètres généraux de TIA Portal à l'aide de TIA Portal Openness :

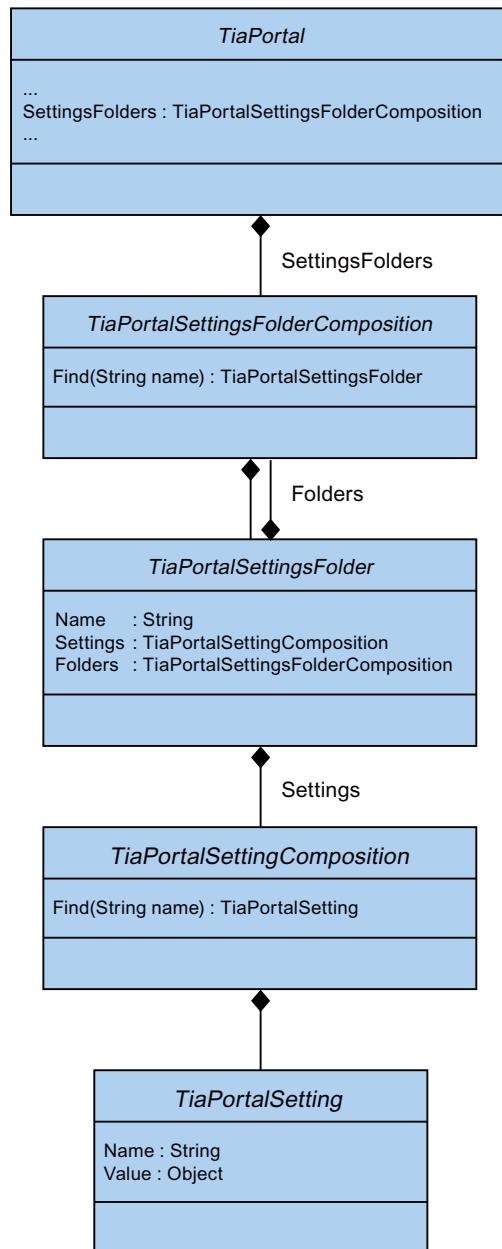
- Langue actuelle de l'interface utilisateur
- Option "Rechercher dans le projet" pour créer l'index de recherche requis dans un projet

Le tableau ci-dessous donne les détails des paramètres accessibles dans la section "Général" des paramètres de TIA Portal. L'instance `TiaPortalSettingsFolder` porte le nom "Général".

Nom	Type de données	Accessible en écriture	Description
"SearchInProject"	System.Boolean	r/w	Active ou désactive la création de l'index de recherche dans un projet.
"UserInterfaceLanguage"	System.CultureInfo	r/w	Affiche la langue active pour l'interface utilisateur de TIA Portal.

L'accès à ces paramètres se fait par la classe `TiaPortalSettingsFolder`. La classe `TiaPortalSettingsFolder` est accessible par l'attribut `Settings` de la classe `TiaPortal`.

La figure ci-dessous montre les paramètres spécifiques dans TIA Portal Openness :



Code de programme : rechercher dans le projet

Pour activer ou désactiver l'option "Rechercher dans le projet", modifiez le code de programme suivant.

```
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;
    }
}
```

Code de programme : langue de l'interface utilisateur

Pour accéder à la langue actuelle de l'interface utilisateur, modifiez le code de programme suivant :

```
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");
    }
}
```

Voir aussi

Hierarchie des objets matériels du modèle d'objet (Page 64)

7.10.4 Accéder à des langues

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

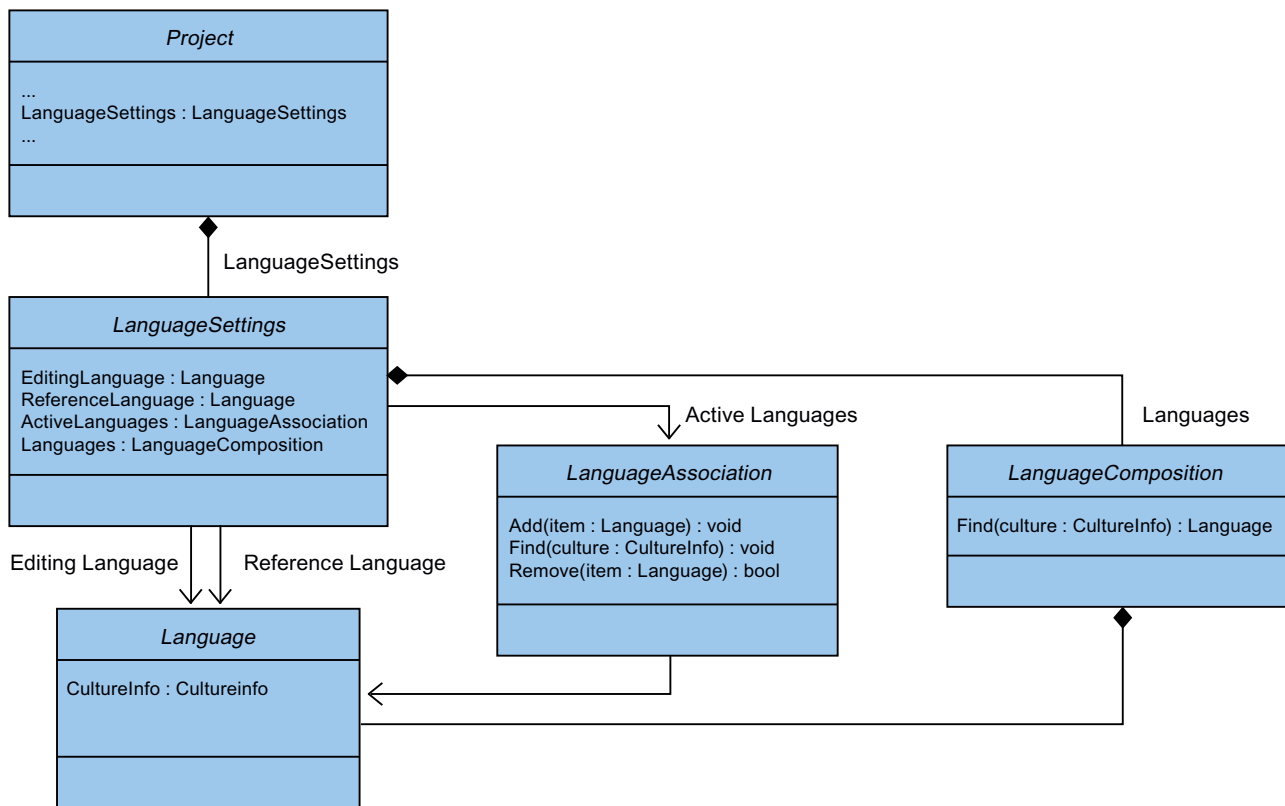
Utilisation

Dans TIA Portal, vous pouvez déterminer la langue du projet et la gérer dans l'éditeur "Langues de projet".

TIA Portal Openness prend en charge l'accès suivant aux langues de projet :

- Itération par langues prises en charge
- Recherche dans le recueil de langues prises en charge à l'aide de `System.Globalization.CultureInfo`
- Accès à certaines langues. Chaque objet langue contient un attribut unique en lecture seule `Culture` du type `System.Globalization.CultureInfo..`
- Accès à un recueil de langues actives
- Recherche dans le recueil de langues actives à l'aide de `System.Globalization.CultureInfo`
- Ajout d'une langue à un recueil de langues actives
- Suppression d'une langue d'un recueil de langues actives
- Choix d'une langue d'édition
- Choix d'une langue de référence

Ces fonctions sont mises à disposition par l'objet `LanguageSettings`. La figure ci-dessous montre le modèle dans TIA Portal Openness :



Code de programme : déterminer une langue

Modifiez le code de programme suivant pour déterminer une langue. Quand vous définissez une langue inactive à l'aide de TIA Portal Openness, elle est ajoutée à la liste des langues actives.

```

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;
  
```

Code de programme : désactiver une langue active

Pour désactiver une langue active, modifiez le code de programme suivant. Quand vous désactivez une langue utilisée comme langue de référence ou d'édition, la langue sélectionnée est cohérente avec le comportement dans l'interface utilisateur.

```
Project project = ...;

LanguageSettings languageSettings = project.LanguageSettings;
LanguageAssociation activeLanguages = languageSettings.ActiveLanguages;
Language activeGermanLanguage = activeLanguages.Find(CultureInfo.GetCultureInfo("de-DE"));
activeLanguages.Remove(activeGermanLanguage);
```

Voir aussi

Hiérarchie des objets matériels du modèle d'objet (Page 64)

7.10.5 Déterminer la structure et les attributs de l'objet

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)

Utilisation

Vous pouvez déterminer la structure de navigation de la hiérarchie d'objet à l'aide de l'interface `IEngineeringObject`. Le résultat est retourné sous forme de liste :

- Objets inférieurs
- Compositions inférieures
- Tous les attributs

Signature

Utilisez la méthode `GetAttributeInfos` pour déterminer les attributs.

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

Code de programme : Déterminer des objets ou compositions

Utilisez le code de programme suivant pour afficher tous les noms de compositions :

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

Si vous connaissez la valeur de retour, modifiez le code de programme suivant :

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

Code de programme : détermination des attributs

Pour fournir en retour les attributs d'un objet aux droits d'accès spécifiques dans une liste, modifiez le code de programme suivant :

```
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;
}
```

7.10.6 Accéder au logiciel cible

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour mettre à disposition un logiciel cible, modifiez le code de programme suivant :

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

Pour accéder aux attributs du logiciel, modifiez le code de programme suivant :

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

7.10.7 Accéder à des textes multilingues et les énumérer

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Des textes multilingues dans TIA Portal sont, par exemple, Project.Comment, PlcTag.Comment, etc. Dans TIA Portal Openness, les textes multilingues sont représentés par l'objet `MultilingualText`. Un objet `MultilingualText` se compose de `MultilingualTextItemComposition`.

`MultilingualTextItemComposition` prend en charge la méthode `Find` suivante :

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

Chaque `MultilingualTextItem` a les attributs suivants :

Nom d'attribut	Type de données	Accessible en écriture	Description
Language	Siemens.Engineering.Language	r/o	Langue de cet élément
Text	System.String	r/w	Texte indiqué pour cette langue

Code de programme : déterminer des textes multilingues

```
...
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";
...
```

7.10.8 Compiler le projet

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Tous les appareils sont "hors ligne".

Utilisation

L'interface API prend en charge la compilation d'appareils et de blocs de programme. Le résultat de la compilation est retourné en tant qu'objet. Selon le type d'objet, la compilation HW, SW ou HW/SW est mise à disposition. Les types d'objet suivants sont pris en charge :

- Device - HW & SW
 - Device avec CPU de de sécurité - SW
- DeviceItem - HW
- CodeBlock - SW
- DataBlock - SW
- HmiTarget - SW
- PlcSoftware - SW
- PlcType - SW
- PlcBlockSystemGroup - SW
- PlcBlockUserGroup - SW
- PlcTypeSystemGroup - SW
- PlcTypeUserGroup - SW

Remarque

Format de l'horodatage

Tous les horodatages sont en UTC. Si vous souhaitez afficher l'heure locale, vous pouvez utiliser `DateTime.ToLocalTime()`.

Signature

Pour la compilation, utilisez la méthode `ICompilable`.

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

Remarque

Tous les appareils doivent être "hors ligne" avant le début de la compilation.

Code de programme

Pour compiler les modifications logicielles d'un objet de type `HmiTarget`, modifiez le code de programme suivant :

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

Pour compiler les modifications logicielles d'un objet de type `PlcSoftware`, modifiez le code de programme suivant :

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

Pour compiler les modifications logicielles d'un objet de type `CodeBlock`, modifiez le code de programme suivant :

```
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();
    }
}
```

Pour évaluer le résultat de la compilation, modifiez le code de programme suivant :

```
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);
    }
}
```

Voir aussi

[Importation de données de configuration \(Page 377\)](#)

7.10.9 Lire des attributs liés au projet

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Cette fonction vous permet d'appeler des attributs liés au projet issus du TIA Portal Openness API. Les informations fournies comprennent les attributs du projet, l'historique du projet et les produits utilisés par le projet.

Attributs du projet

Les attributs du projet fournissent les informations suivantes :

Nom d'attribut	Type de données	Accessible en écriture	Description
Author	System.String	r/o	Auteur du projet.
Comment	Siemens.Engineering.MultilingualText	r/o	Commentaire du projet.
Copyright	System.String	r/o	Mention de copyright du projet.
CreationTime	System.DateTime	r/o	Date et heure auxquelles le projet a été créé.
Family	System.String	r/o	Famille du projet.
IsModified	System.Boolean	r/o	Affiche vrai si le projet a été modifié.
LanguageSettings	Siemens.Engineering.LanguageSettings	r/o	Gère les langues du projet.
LastModified	System.DateTime	r/o	Date et heure auxquelles le projet a été modifié pour la dernière fois.
LastModifiedBy	System.String	r/o	Auteur de la dernière modification.
Name	System.String	r/o	Nom du projet.
Path	System.IO.FileInfo	r/o	Chemin absolu du projet.
Size	System.Int64	r/o	Taille du projet, en Ko.
Version	System.String	r/o	Version du projet.

Pour accéder aux attributs liés au projet, modifiez le code de programme suivant :

```
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;
```

Pour énumérer les langues du projet, modifiez le code de programme suivant :

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

Pour obtenir le texte de commentaire, modifiez le code de programme suivant :

```
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;
```

Historique du projet

L'historique du projet regroupe des objets du type `HistoryEntry`, qui contiennent les informations suivantes :

Nom d'attribut	Type de données	Accessible en écriture	Description
Texte	System.String	r/o	Description de l'événement.
DateTime	System.DateTime	r/o	Date et heure auxquelles l'événement s'est produit.

Pour énumérer les `HistoryEntries` et accéder à leurs attributs, modifiez le code de programme suivant :

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

Remarque

L'attribut `Texte` de `HistoryEntry` contient une chaîne de caractères dans la même langue que l'interface utilisateur. Lorsqu'une application TIA Portal Openness est attachée à un TIA Portal sans interface utilisateur, la chaîne de caractères est fournie en anglais par défaut.

Produits utilisés

L'objet `UsedProduct` contient les informations suivantes :

Nom d'attribut	Type de données	Accessible en écriture	Description
Nom	System.String	r/o	Nom du projet utilisé.
Version	System.String	r/o	Version du projet.

Pour énumérer `UsedProduct` et accéder aux attributs, modifiez le code de programme suivant :

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

7.10.10 Suppression d'un graphique du projet

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour supprimer une bibliothèque de graphiques, modifiez le code de programme suivant :

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

7.10.11 Enregistrer le projet

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Utilisez la méthode `project.Save()` pour enregistrer un projet.

Code du programme

Pour ouvrir et enregistrer un projet, modifiez le code de programme suivant :

```
public static void SaveProject(TiaPortal tiaPortal)
{
    Project project = null;
    //Use the code in the try block to open and save a project
    try
    {
        project = tiaPortal.Projects.Open(new FileInfo(@"Some\Path\MyProject.ap14"));
        //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();
    }
}
```

7.10.12 Fermer un projet

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour fermer un projet, modifiez le code de programme suivant :

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


7.11 Fonctions pour connexions

7.11.1 Attributs configurables d'une liaison port à port

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Ouvrir un projet (Page 99)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les attributs d'une interconnexion de ports se trouvent dans l'élément d'appareil port. L'accès en lecture et en écriture des attributs avec TIA Portal Openness est identique à celui dans l'interface utilisateur.

Paramètres du port

Les attributs suivants sont disponibles pour paramétrer l'interface du port :

Nom d'attribut	Type de données	Accessible en écriture	Accès	Description
MediumAttachmentType	MediumAttachmentType	r/o	Attribut dynamique	
CableName	CableName	r/w	Attribut dynamique	
AlternativePartnerPorts	Bool	r/w	Attribut dynamique	Disponible uniquement quand la fonction de changeur d'outil est prise en charge, par ex. pour la CPU1516.
SignalDelaySelection	SignalDelaySelection	r/w	Attribut dynamique	
CableLength	CableLength	r/w	Attribut dynamique	
SignalDelayTime	Double	r/w	Attribut dynamique	

Les valeurs ENUM suivantes sont disponibles pour l'attribut `MediumAttachmentType` :

Valeur	Description
<code>MediumAttachmentType.None</code>	Type de couplage impossible à déterminer.
<code>MediumAttachmentType.Copper</code>	Couplage cuivre.
<code>MediumAttachmentType.FibreOptic</code>	Couplage fibre optique.

Les valeurs ENUM suivantes sont disponibles pour l'attribut `CableName` :

Valeur	Description
<code>CableName.None</code>	Aucun nom de câble indiqué
<code>CableName.FO_Standard_Cable_9</code>	Câble standard FO GP (9 µm)
<code>CableName.Flexible_FO_Cable_9</code>	Câble FO souple (9 µm)
<code>CableName.FO_Standard_Cable_GP_50</code>	Câble standard FO GP (50 µm)
<code>CableName.FO_Trailing_Cable_GP</code>	Câble traînant FO / GP
<code>CableName.FO_Ground_Cable</code>	Conducteur de terre FO
<code>CableName.FO_Standard_Cable_62_5</code>	Câble standard FO (62,5 µm)
<code>CableName.Flexible_FO_Cable_62_5</code>	Câble FO souple (62,5 µm)
<code>CableName.POF_Standard_Cable_GP</code>	Câble standard POF GP
<code>CableName.POF_Trailing_Cable</code>	Câble traînant POF
<code>CableName.PCF_Standard_Cable_GP</code>	Câble standard POF GP
<code>CableName.PCF_Trailing_Cable_GP</code>	Câble traînant PCF / GP
<code>CableName.GI_POF_Standard_Cable</code>	Câble standard POF GI
<code>CableName.GI_POF_Trailing_Cable</code>	Câble traînant POF GI
<code>CableName.GI_PCF_Standard_Cable</code>	Câble standard PCF GI
<code>CableName.GI_PCF_Trailing_Cable</code>	Câble traînant PCF GI

Les valeurs ENUM suivantes sont disponibles pour l'attribut `SignalDelaySelection` :

Valeur	Description
<code>SignalDelaySelection.None</code>	
<code>SignalDelaySelection.CableLength</code>	<code>CableLength</code> sert à déterminer le retard du signal.
<code>SignalDelaySelection.SignalDelayTime</code>	<code>CableDelayTime</code> sert à déterminer le retard du signal.

Les valeurs ENUM suivantes sont disponibles pour l'attribut `CableLength` :

Valeur	Description
<code>CableLength.None</code>	La longueur de câble n'est pas indiquée
<code>CableLength.Length20m</code>	Longueur de câble 20 m.
<code>CableLength.Length50m</code>	Longueur de câble 50 m.
<code>CableLength.Length100m</code>	Longueur de câble 100 m.
<code>CableLength.Length1000m</code>	Longueur de câble 1000 m.
<code>CableLength.Length3000m</code>	Longueur de câble 3000 m.

Options de port

Les attributs suivants sont disponibles pour les options de port :

Nom d'attribut	Type de données	Acces- sible en écriture	Accès
<code>PortActivation</code>	Bool	r/w	Attribut dynamique
<code>TransmissionRateAndDuplex</code>	<code>TransmissionRateAndDuplex</code>	r/w	Attribut dynamique

Nom d'attribut	Type de données	Acces- sible en écriture	Accès
PortMonitoring	Bool	r/w	Attribut dynamique
TransmissionRateAutoNegotiation	Bool	r/w	Attribut dynamique
EndOfDetectionOfAccessibleDevices	Bool	r/w	Attribut dynamique
EndOfTopologyDiscovery	Bool	r/w	Attribut dynamique
EndOfSyncDomain	Bool	r/w	Attribut dynamique

Les valeurs ENUM suivantes sont disponibles pour l'attribut TransmissionRateAndDuplex :

Valeur	Description
TransmissionRateAndDuplex.None	
TransmissionRateAndDuplex.Automatic	Automatique
TransmissionRateAndDuplex.AUI10Mbps	AUI 10 Mbps
TransmissionRateAndDuplex.TP10MbpsHalfDuplex	TP 10 Mbps semi-duplex
TransmissionRateAndDuplex.TP10MbpsFullDuplex	TP 10 Mbps duplex intégral
TransmissionRateAndDuplex.AsyncFiber10MbpsHalfDuplex	Fibre optique asynchrone 10 Mbps semi-duplex
TransmissionRateAndDuplex.AsyncFiber10MbpsFullDuplex	Fibre optique asynchrone 10 Mbps duplex intégral
TransmissionRateAndDuplex.TP100MbpsHalfDuplex	TP 100 Mbps semi-duplex
TransmissionRateAndDuplex.TP100MbpsFullDuplex	TP 100 Mbps duplex intégral
TransmissionRateAndDuplex.FO100MbpsFullDuplex	FO 100 Mbps duplex intégral
TransmissionRateAndDuplex.X1000MbpsFullDuplex	X1000 Mbps duplex intégral
TransmissionRateAndDuplex.FO1000MbpsFullDuplexLD	FO 1000 Mbps duplex intégral LD
TransmissionRateAndDuplex.FO1000MbpsFullDuplex	FO 1000 Mbps duplex intégral
TransmissionRateAndDuplex.TP1000MbpsFullDuplex	TP 1000 Mbps duplex intégral
TransmissionRateAndDuplex.FO10000MbpsFullDuplex	FO 10000 Mbps duplex intégral
TransmissionRateAndDuplex.FO100MbpsFullDuplexLD	FO 100 Mbps duplex intégral LD
TransmissionRateAndDuplex.POFPCF100MbpsFullDuplexLD	POF/PCF 100 Mbps duplex intégral

Voir aussi

Etablissement d'une connexion au portail TIA (Page 74)

7.12 Fonctions pour bibliothèques

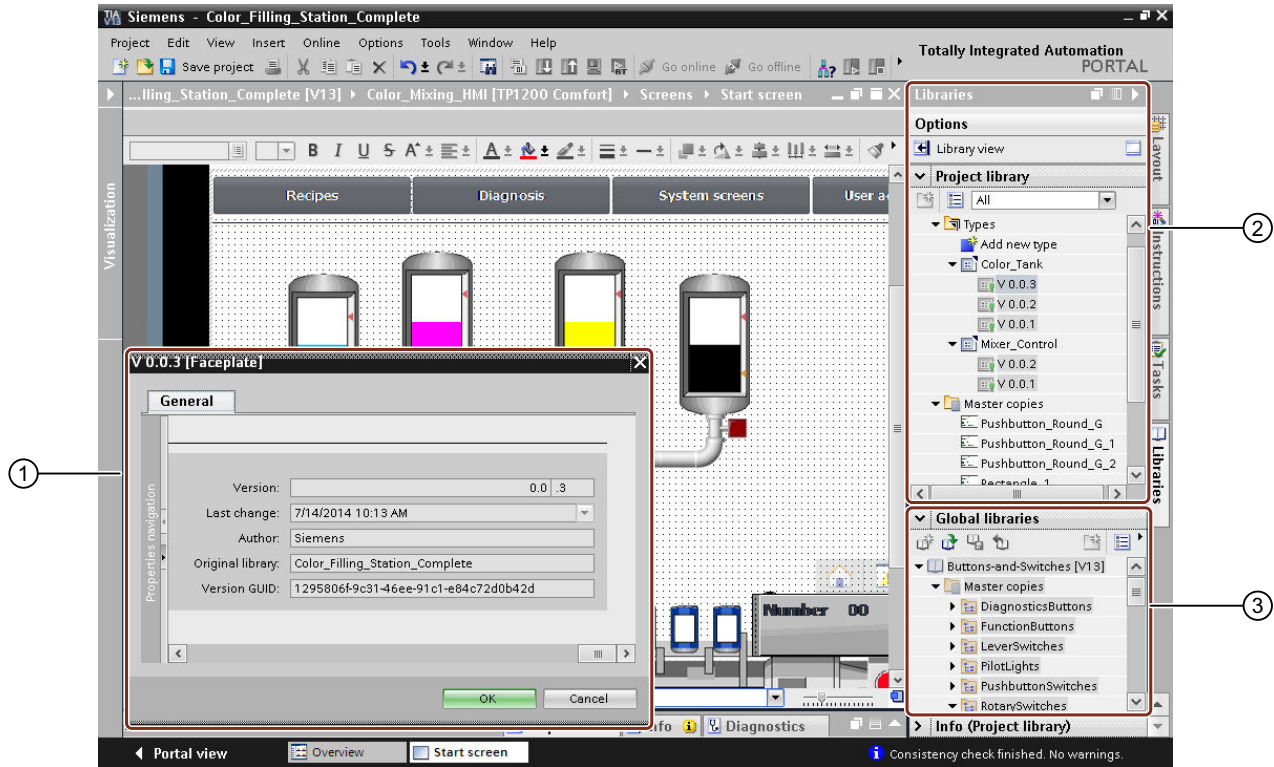
7.12.1 Fonctions pour objets et instances

Accès aux types et instances

L'interface TIA Portal Openness API vous permet d'accéder aux types, versions de types et copies maîtres dans la bibliothèque du projet ou les bibliothèques globales. Vous pouvez déterminer des liaisons entre les versions de types et les instances. Vous pouvez également actualiser les instances dans le projet et synchroniser les modifications entre une bibliothèque globale et la bibliothèque de projet. En outre, l'interface TIA Portal Openness API prend en charge la comparaison des versions de type et des instances.

Fonctions pour des objets et des instances

L'interface TIA Portal Openness API vous permet d'accéder aux fonctions suivantes pour les types, versions de types, copies maîtres et instances :



- ① Afficher les attributs de types, versions de types, copies maîtres et instances
- ② Les fonctions suivantes sont disponibles dans la bibliothèque de projet :
 - Mettre à jour les instances des types
 - Instancier les versions de types dans le projet
 - Naviguer à l'intérieur d'un groupe de bibliothèques
 - Supprimer un groupe, des types, des versions de types et des copies maîtres
- ③ Les fonctions suivantes sont disponibles dans la bibliothèque globale :
 - Mettre à jour les instances des types
 - Instancier la version de type dans le projet
 - Naviguer à l'intérieur d'un groupe de bibliothèques

7.12.2 Accès aux bibliothèques globales

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal. Voir Etablissement d'une connexion au portail TIA (Page 74)

Utilisation

Il existe trois types de bibliothèques globales.

- Bibliothèque système globale : ces bibliothèques globales font partie de l'installation de TIA Portal et utilisent l'extension de fichier *.as14. Toutes les bibliothèques système globales sont protégées en écriture.
- Bibliothèque d'entreprise globale : il s'agit de bibliothèques globales qui ont été sélectionnées par l'administrateur et qui sont chargées préalablement au démarrage de TIA Portal. Toutes les bibliothèques d'entreprise globales sont protégées en écriture.
- Bibliothèque utilisateur globale : il s'agit de bibliothèques globales qui ont été créées par l'utilisateur de TIA Portal. Les bibliothèques d'entreprise globales peuvent être protégées en écriture ou accessibles avec des droits en lecture et en écriture.
Si une bibliothèque utilisateur globale est déjà ouverte dans un mode donné, elle ne peut pas être ouverte dans un autre mode.
Les bibliothèques utilisateur globales des versions précédentes ne peuvent être ouvertes qu'en mode protégé en écriture.

En plus, les bibliothèques globales ouvertes avec TIA Portal Openness sont ajoutées au recueil des bibliothèques globales de l'interface utilisateur TIA Portal et affichées dans l'interface dans TIA Portal tant que celle-ci est présente.

Code de programme : bibliothèques globales disponibles

Pour appeler des informations sur toutes les bibliothèques globales, modifiez le code de programme suivant :

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

Attributs de GlobalLibrary

Valeur	Type de données	Description
Author	String	Auteur de la bibliothèque globale
Comment	MultilingualText	Commentaire sur la bibliothèque globale
IsReadOnly	Boolean	Vrai si la bibliothèque globale est protégée en écriture.
IsModified	Boolean	Vrai si le contenu de la bibliothèque globale a été modifié.

Valeur	Type de données	Description
Name	String	Nom de la bibliothèque globale
Path	FileInfo	Spécification du chemin de la bibliothèque globale

Attributs de GlobalLibraryInfo

Valeur	Type de sortie	Description
IsReadOnly	Boolean	Vrai si la bibliothèque globale est protégée en écriture.
IsOpen	Boolean	Vrai si la bibliothèque globale est déjà ouverte.
LibraryType	GlobalLibraryType	Type de la bibliothèque globale : <ul style="list-style-type: none"> • System : bibliothèque système globale • Corporate : bibliothèque d'entreprise globale • User : bibliothèque utilisateur globale
Name	String	Nom de la bibliothèque globale
Path	FileInfo	Spécification du chemin de la bibliothèque globale

Voir aussi

Accéder aux dossiers dans une bibliothèque (Page 132)

7.12.3 Ouvrir des bibliothèques

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness. Cette condition s'applique uniquement à l'accès aux bibliothèques de projet.
Voir Ouvrir un projet (Page 99)

Utilisation

Vous pouvez ouvrir une bibliothèque globale via System.IO.FileInfo par le biais d'un chemin d'accès au fichier de bibliothèque sur un support d'enregistrement local ou un dispositif de stockage réseau. Seules les bibliothèques utilisateur globales peuvent être ouvertes par le biais d'un chemin d'accès. Vous ne pouvez pas utiliser le chemin d'accès d'une bibliothèque système globale ou d'une bibliothèque d'entreprise globale pour ouvrir une bibliothèque.

A partir de V14 SP1, GlobalLibraryInfo permet d'ouvrir des bibliothèques globales. OpenMode est indiquée dans GlobalLibraryInfo.

Vous pouvez mettre à niveau une bibliothèque utilisateur globale issue d'une version antérieure de TIA Portal et l'ouvrir avec la version actuelle de TIA Portal. Une mise à niveau

ne permet pas d'ouvrir une bibliothèque globale issue de V13 ou d'une version antérieure. Ces bibliothèques doivent d'abord être mises à jour à V13 SP1.

En plus, les bibliothèques globales ouvertes avec TIA Portal Openness sont ajoutées au recueil des bibliothèques globales de TIA Portal et affichées dans l'interface utilisateur de TIA Portal.

Code de programme : ouvrir une bibliothèque au moyen de System.IO.FileInfo

Modifiez le code de programme suivant :

```
TiaPortal tia = ...
FileInfo fileInfo = ....

UserGlobalLibrary userLib = tia.GlobalLibraries.Open(fileInfo, OpenMode.ReadWrite);
```

Code de programme : ouvrir une bibliothèque au moyen de GlobalLibraryInfo

Modifiez le code de programme suivant :

```
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);
```

Code de programme : mettre une bibliothèque à niveau

Modifiez le code de programme suivant :

```
TiaPortal tia = ...
FileInfo fileInfo = .... //library from previous TIA Portal version

UserGlobalLibrary userLib = tia.GlobalLibraries.OpenWithUpgrade(fileInfo);
```

OpenMode

Valeur	Description
ReadOnly	Accès en écriture à la bibliothèque.
ReadWrite	Accès en lecture et écriture à la bibliothèque.

7.12.4 Énumérer des bibliothèques ouvertes

Condition requise

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)

Utilisation

Vous pouvez énumérer toutes les bibliothèques globales ouvertes dans TIA Portal, qu'elles aient été ouvertes via l'API ou l'interface utilisateur.

Les bibliothèques globales issues de versions antérieures de TIA Portal ne sont pas énumérées lorsqu'elles sont ouvertes avec accès en écriture.

Code du programme

Pour énumérer les bibliothèques globales, modifiez le code de programme suivant :

```
TiaPortal tia = ...
foreach (GlobalLibrary globLib in tia.GlobalLibraries)
{
    ///work with the global library
}
```

Voir aussi

Ouvrir un projet (Page 99)

7.12.5 Enregistrer et fermer les bibliothèques

Condition

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Une bibliothèque est ouverte.
Voir Ouvrir des bibliothèques (Page 128)

Utilisation

Vous pouvez fermer ou enregistrer des bibliothèques utilisateur globales. Toutes les modifications apportées à la bibliothèque globale ne sont pas enregistrées automatiquement. Toutes les modifications non enregistrées sont rejetées sans invite utilisateur lorsqu'une bibliothèque globale est fermée.

Vous ne pouvez fermer ou enregistrer des bibliothèques système globales ni des bibliothèques d'entreprise globales.

Code du programme

Modifiez le code de programme suivant :

```
UserGlobalLibrary userLib = ...

// close and discard changes
userLib.Close();

// save changes and close library
userLib.Save();
userLib.Close();
```

7.12.6 Créer des bibliothèques globales

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)

Utilisation

La TIA Portal Openness API permet de créer des bibliothèques globales par appel de la méthode `Create` sur `GlobalLibraryComposition`. Une bibliothèque utilisateur globale est affichée.

`GlobalLibraryComposition.Create`

Modifiez le code de programme suivant :

```
TiaPortal tia= ...;
DirectoryInfo targetDirectory = new DirectoryInfo(@"D:\GlobalLibraries");
UserGlobalLibrary globalLibrary =
tia.GlobalLibraries.Create<UserGlobalLibrary>(targetDirectory, "Library1")
```

Dans l'exemple utilisé,

- un dossier "D:\GlobalLibraries\Library1" est créé
- un fichier de bibliothèque globale "D:\GlobalLibraries\Library1\Library1.alXX" est créé

Paramètres pour créer des bibliothèques globales

Paramètre	Type de données	Type	Description
Author	String	Obligatoire	Auteur d'une bibliothèque globale.
Comment	String	Optionnel	Commentaire d'une bibliothèque globale.
Name	String	Optionnel	Nom d'une bibliothèque globale.
TargetDirectory	DirectoryInfo	Obligatoire	Répertoire dans lequel sera placé le dossier contenant la bibliothèque globale.

Voir aussi

Ouvrir un projet (Page 99)

7.12.7 Accéder aux dossiers dans une bibliothèque

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)
- Vous avez accès à la bibliothèque requise.
Voir Accès aux bibliothèques globales (Page 126).

Utilisation

L'interface TIA Portal Openness API vous permet d'accéder aux dossiers système pour les types et copies maîtres dans une bibliothèque. Vous pouvez alors accéder aux types, versions de types, copies maîtres et dossiers personnalisés dans le dossier système.

La méthode `Find`, par ex.

```
libTypeUserFolder.Folders.Find("SomeUserFolder");
```

vous permet d'accéder à tout moment à un dossier personnalisé.

Code de programme : accéder au dossier système

Pour accéder au dossier système pour les types dans une bibliothèque, modifiez le code de programme suivant :

```
public static void AccessTypeSystemFolder(ILibrary library)
{
    LibraryTypeSystemFolder libTypeSystemFolder = library.TypeFolder;
}
```

Pour accéder au dossier système pour les copies maîtres dans une bibliothèque, modifiez le code de programme suivant :

```
public static void AccessMasterCopySystemFolder(ILibrary library)
{
    MasterCopySystemFolder libMasterCopySystemFolder = library.MasterCopyFolder;
}
```

Code de programme : accéder à des dossiers personnalisés avec la méthode Find()

Modifiez le code de programme suivant :

```
...
LibraryTypeUserFolderComposition userFolderComposition = ...
LibraryTypeUserFolder userFolder = userFolderComposition.Find("Name of user folder");
...
```

Code de programme : énumérer des dossiers personnalisés

Pour énumérer les sous-dossiers personnalisés dans un dossier système pour des types, modifiez le code de programme suivant :

```
public static void EnumerateUserFoldersInTypeSystemFolder(ILibrary library)
{
    // Enumerating user folders in type system folder:
    LibraryTypeSystemFolder libTypeSystemFolder = library.TypeFolder;
    foreach (LibraryTypeUserFolder libTypeUserFolder in libTypeSystemFolder.Folders)
    {
        //...
    }
}
```

Pour énumérer les sous-dossiers personnalisés dans un dossier système pour des copies maîtres, modifiez le code de programme suivant :

```
public static void EnumerateUserFoldersInMasterCopySystemFolder(ILibrary library)
{
    // Enumerating user folders in master copy system folder:
    MasterCopySystemFolder libMasterCopySystemFolder = library.MasterCopyFolder;
    foreach (MasterCopyUserFolder libMasterCopyUserFolder in
libMasterCopySystemFolder.Folders)
    {
        //..
    }
}
```

Pour énumérer les sous-dossiers personnalisés dans un dossier personnalisé pour des types, modifiez le code de programme suivant :

```
public static void EnumerateAllUserFolders(LibraryTypeUserFolder libUserFolder)
{
    foreach (LibraryTypeUserFolder libSubUserFolder in libUserFolder.Folders)
    {
        EnumerateAllUserFolders(libSubUserFolder);
    }
}
```

Pour énumérer les sous-dossiers personnalisés dans un dossier personnalisé pour des copies maîtres, modifiez le code de programme suivant :

```
public static void EnumerateAllUserFolders(MasterCopyUserFolder libUserFolder)
{
    foreach (MasterCopyUserFolder libSubUserFolder in libUserFolder.Folders)
    {
        EnumerateAllUserFolders(libSubUserFolder);
    }
}
```

Code de programme : créer des dossiers personnalisés

Pour créer un dossier personnalisé pour des types, modifiez le code de programme suivant :

```
var typeFolderComposition = ProjectLibrary.TypeFolder.Folders;
var newTypeUserFolder = typeFolderComposition.Create("NewTypeUserFolder");
```

Pour créer un dossier personnalisé pour des copies maîtres, modifiez le code de programme suivant :

```
var masterCopyFolderComposition = projectProjectLibrary.MasterCopyFolder.Folders;
MasterCopyUserFolder newMasterCopyUserFolder =
masterCopyFolderComposition.Create("NewMasterCopyUserFolder");
```

Code de programme : renommer des dossiers personnalisés

Pour créer un dossier personnalisé pour des types, modifiez le code de programme suivant :

```
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")});
```

Pour créer un dossier personnalisé pour des copies maîtres, modifiez le code de programme suivant :

```
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")});
```

Voir aussi

[Accéder à des modèles de copie \(Page 144\)](#)

7.12.8 Accéder aux types

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir [Etablissement d'une connexion au portail TIA \(Page 74\)](#)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir [Ouvrir un projet \(Page 99\)](#)
- Vous avez accès à la bibliothèque requise.
Voir [Accès aux bibliothèques globales \(Page 126\)](#).
- Vous avez accès à un groupe pour les types.
Voir [Accéder aux dossiers dans une bibliothèque \(Page 132\)](#).

Utilisation

Vous pouvez accéder aux types d'une bibliothèque par le biais de l'interface TIA Portal Openness API.

- Vous pouvez énumérer les types.
- Vous pouvez renommer les types.
- Vous pouvez accéder aux attributs suivants des différents types :

Attribut	type de données	Description
Author	String	Fournit le nom de l'auteur :
Comment	MultilingualText	Fournit en retour le commentaire.
Guid	Guid	Fournit en retour le GUID du type. ¹
Name	String	Fournit en retour le nom du type. ²

¹ Cet attribut vous permet de trouver un type précis dans une bibliothèque. La recherche est récursive.

² Cet attribut vous permet de trouver un type précis dans un dossier. Les sous-dossiers ne sont pas pris en compte dans la recherche. Les noms des types ne sont pas univoques. Plusieurs types avec le même nom peuvent exister dans différents groupes. Le GUID du type est en revanche univoque.

Sous-classes pour objets de types de bibliothèques

La TIA Portal Openness permet d'accéder à des objets de types de bibliothèque via des sous-classes. Les sous-classes suivantes sont disponibles :

- 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

Le code suivant est un exemple d'utilisation de sous-classes pour types de bibliothèques.

```
ProjectLibrary library = project.ProjectLibrary;
VBScriptLibraryType vbScriptType = ...;

VBScriptLibraryType libraryTypeAsVbScript = libraryType as VBScriptLibraryType;
```


Code du programme

Pour énumérer tous les types dans le dossier système d'une bibliothèque, modifiez le code de programme suivant :

```
public static void EnumerateTypesInTypesSystemFolder (LibraryTypeSystemFolder
libraryTypeSystemFolder)
{
    foreach (LibraryType libraryType in libraryTypeSystemFolder.Types)
    {
        //...
    }
}
```

Pour énumérer tous les types dans le dossier personnalisé d'une bibliothèque, modifiez le code de programme suivant :

```
public static void EnumerateTypesInTypesUserFolder (LibraryTypeUserFolder
libraryTypeUserGroup)
{
    foreach (LibraryType libraryType in libraryTypeUserGroup.Types)
    {
        //...
    }
}
```

Pour accéder aux attributs d'un type, modifiez le code de programme suivant :

```
public static void InspectPropertiesOfType (LibraryType libTypeObject)
{
    string typeAuthor = libTypeObject.Author;
    MultilingualText typeComment = libTypeObject.Comment;
    string typeName = libTypeObject.Name;
    Guid typeGUID = libTypeObject.Guid;
}
```

Pour trouver un type précis par son nom ou son GUID, modifiez le code de programme suivant :

```
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");
}
```

Pour renommer un type, modifiez le code de programme suivant :

```
// 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")});
```

7.12.9 Accéder aux types de versions

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)
- Vous avez accès à la bibliothèque requise.
Voir Accès aux bibliothèques globales (Page 126).
- Vous avez accès à un groupe pour types.
Voir Accéder aux dossiers dans une bibliothèque (Page 132).

Utilisation

L'interface TIA Portal Openness API vous donne accès aux versions de type.

- Vous pouvez énumérer les versions d'un type.
- Vous pouvez déterminer le type auquel une version appartient.
- Vous pouvez énumérer les instances d'une version de type.
- Vous pouvez créer une nouvelle instance d'une version de type.
- Vous pouvez naviguer d'une instance vers la version d'objet reliée à celle-ci.
- Vous pouvez accéder aux attributs suivants des différentes versions de type :

Attribut	Type de données	Description
Author	String	Fournit le nom de l'auteur :
Comment	MultilingualText	Fournit en retour le commentaire.
Guid	Guid	Fournit en retour le GUID de la version de type. ¹
ModifiedDate	DateTime	Fournit en retour la date et l'heure à laquelle la version de type a été mise à l'état "Committed".

Attribut	Type de données	Description
State	LibraryTypeVersionState	Fournit en retour l'état de la version : <ul style="list-style-type: none"> • <code>InWork</code> : correspond, selon le type affecté, à l'état "En cours" ou "Test en cours". • <code>Committed</code> : correspond à l'état "Validé".
TypeObject	LibraryType	Fournit en retour le type auquel appartient cette version de type.
VersionNumber	Version	Fournit en retour le numéro de version sous forme de désignation à trois chiffres, par ex. "1.0.0". ²

¹ Cet attribut vous permet de trouver une version de type spécifique dans une bibliothèque.

² Cet attribut vous permet de trouver une version de type spécifique dans une composition "LibraryTypeVersion".

Énumérer les versions d'un type

Modifiez le code de programme suivant :

```
//Enumerate the type versions of a type
public static void EnumerateVersionsInType(LibraryType libraryType)
{
    foreach (LibraryTypeVersion libraryTypeVersion in libraryType.Versions)
    {
        //...
    }
}
```

Appeler les attributs d'une version de type

Modifiez le code de programme suivant :

```
//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;
}
```

Créer une instance d'une version de type

Vous pouvez créer une nouvelle instance d'une version de type. Les objets suivants sont pris en charge :

- Blocs (FB/FC)
- Types de données personnalisé API
- Vues
- Scripts VB

Une instance d'une version de type peut être créée à partir de la bibliothèque globale ou de la bibliothèque de projet. Si vous créez une instance d'une version de type depuis une bibliothèque globale, la version de type est d'abord synchronisée avec la bibliothèque de projet.

Une exception récupérable est déclenchée lorsqu'il n'est pas possible de créer une instance dans la cible. Causes possibles :

- La version de type de bibliothèque est en cours de traitement
- Une instance de la version de type de bibliothèque existe déjà dans l'appareil cible

Modifiez le code de programme suivant :

```
VBScriptLibraryTypeVersion scriptVersion = ...;
VBScriptComposition vbscripts = ...;

//Using the CreateFrom method to create an instance of the version in the VBScripts
composition
VBScript newScript = vbscripts.CreateFrom(scriptVersion);
```

Modifiez le code de programme suivant :

```
ScreenLibraryTypeVersion screenVersion = ...;
ScreenComposition screens = ...;

//Using the CreateFrom method to create an instance of the version in the screens
composition
Screen newScreen = screens.CreateFrom(screenVersion);
```

Modifiez le code de programme suivant :

```
CodeBlockLibraryTypeVersion blockVersion = ...;
PlcBlockComposition blocks = ...;

//Using the CreateFrom method to create an instance of the version in the blocks composition
PlcBlock newBlock = blocks.CreateFrom(blockVersion);
```

Modifiez le code de programme suivant :

```
PlcTypeLibraryTypeVersion plcTypVersion=...;
PlcTypeComposition types=...;

//Using the CreateFrom method to create an instance of the version in the types composition
PlcType newType = types.CreateFrom(plcTypeVersion);
```

Déterminer les utilisations d'une version de type

Les utilisations suivantes sont différenciées pour les versions de type :

- La version de type utilise d'autres versions de types issues de la bibliothèque.
Exemple : un type de données personnalisé est utilisé dans un bloc de programme. Le bloc de programme doit avoir accès au type de données utilisateur. Cela signifie que le bloc de programme dépend du type de données utilisateur.
Si vous accédez à l'attribut d'indépendance de `CodeBlockLibraryVersion` avec la méthode `GetDependencies()`, vous obtenez une liste de `LibraryTypeVersions` en retour.
- Le type est utilisé par une autre version de type dans la bibliothèque.
Exemple : un type de données personnalisé est utilisé dans un bloc de programme. Le bloc de programme doit avoir accès au type de données utilisateur. Le type de données utilisateur a le bloc de programme correspondant. Le bloc de programme dépend du type de données utilisateur.
Si vous accédez à l'attribut d'indépendance de `PlcTypeLibraryTypeVersion` avec la méthode `GetDependents()`, vous obtenez une liste de `LibraryTypeVersions` en retour.

Une liste contenant des objets du type `LibraryTypeVersion` est fournie en retour pour les deux attributs. Si aucune utilisation n'existe, une chaîne vide est fournie en retour.

Remarque

Une exception peut être déclenchée si vous appliquez ces attributs aux versions de type avec l'état "InWork".

Modifiez le code de programme suivant :

```
//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();
}
```

Code de programme

Pour déterminer le type auquel appartient une version de type, modifiez le code de programme suivant :

```
public static void GetParentTypeOfVersion(LibraryTypeVersion libTypeVersion)
{
    LibraryType parentType = libTypeVersion.TypeObject;
}
```

Pour déterminer les copies maîtres contenant des instances d'une version de type, modifiez le code de programme suivant :

```
public static void GetMasterCopiesContainingInstances(LibraryTypeVersion libTypeVersion)
{
    MasterCopyAssociation masterCopies = libTypeVersion.MasterCopiesContainingInstances;
}
```

Pour trouver une version de type précise par son numéro de version, modifiez le code de programme suivant :

```
public static void FindVersionInLibrary(ILibrary library, Guid versionGUID)
{
    LibraryTypeVersion libTypeVersionByVersionNumber = library.FindVersion(versionGUID);
}
```

7.12.10 Accéder aux instances

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)
- Vous avez accès à la bibliothèque requise.
Voir Accès aux bibliothèques globales (Page 126).
- Vous avez accès à un groupe pour types.
Voir Accéder aux dossiers dans une bibliothèque (Page 132).

Utilisation

L'interface TIA Portal Openness API vous donne accès aux instances de versions de type.

Vous pouvez déterminer toutes les instances d'une version de type par la méthode `FindInstances(IInstanceSearchScope searchScope)`.

Le paramètre `searchScope` vous permet d'indiquer le champ de recherche au sein du projet. Les classes suivantes implémentent l'interface `IInstanceSearchScope` et peuvent être utilisées pour la recherche d'instances :

- `PlcSoftware`
- `HmiTarget`

Cette méthode fournit en retour une liste comprenant des objets du type `LibraryTypeInfo`. Si aucune instance n'existe, une chaîne vide est fournie en retour.

Remarque

Les instances des blocs d'affichage et des types de données utilisateur HMI sont toujours associées à la version de type correspondante.

Les instances de tous les autres objets tels que les blocs de programme ou les vues peuvent être associées à une version de type.

Énumérer les instances d'une version de type

Modifiez le code de programme suivant :

```
//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);
```

Naviguer d'une instance vers la version d'objet reliée à celle-ci

Utilisez l'attribut `LibraryTypeVersion` du service `LibraryTypeInfo` pour naviguer d'une instance vers la version d'objet reliée à celle-ci.

Les objets suivants offrent le service `LibraryTypeInfo` :

- Blocs FB
- Blocs FC
- Types de données personnalisé API
- Vues
- Scripts VB

Lorsque l'objet d'une instance n'est pas relié à une version d'objet, il ne propose pas le service "LibraryTypeInstanceInfo".

```
FC fc = ...;
//Using LibraryTypeInstanceInfo service

LibraryTypeInstanceInfo instanceInfo = fc.GetService<LibraryTypeInstanceInfo>();
if(instanceInfo != null)
{
    LibraryTypeVersion connectedVersion = instanceInfo.LibraryTypeVersion;
    FC parentFc = instanceInfo.LibraryTypeInstance as FC; //parentFc == fc
}
```

Code du programme

Modifiez le code de programme suivant en :

7.12.11 Accéder à des modèles de copie

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Vous avez accès à la bibliothèque requise.
Voir Accès aux bibliothèques globales (Page 126)
- Vous avez accès à un groupe pour les copies maîtres.
Voir Accéder aux dossiers dans une bibliothèque (Page 132)

Utilisation

L'interface TIA Portal Openness API prend en charge l'accès aux copies maîtres dans une bibliothèque globale et la bibliothèque de projet :

- Créer des copies maîtres
- Enumérer des copies maîtres dans des dossiers système et des dossiers personnalisés
- Renommer des copies maîtres
- Interroger les informations des copies maîtres
- Interroger les informations des objets dans une copie maître

Attribut	type de données	Description
Author	String	Fournit le nom de l'auteur :
ContentDescriptions	MasterCopyContentDescriptionComposition	Fournit en retour une description du contenu de la copie maître.
CreationDate	DateTime	Fournit en retour la date de création.
Name	String	Fournit en retour le nom de la copie maître.

Code du programme

Pour énumérer toutes les copies maître dans le dossier système d'une bibliothèque, modifiez le code de programme suivant :

```
public static void EnumerateMasterCopiesInSystemFolder
(MasterCopySystemFolder masterCopySystemFolder)
{
    foreach (MasterCopy masterCopy in masterCopySystemFolder.MasterCopies)
    {
        //...
    }
}
```

Pour accéder à une copie maître individuelle par la méthode "Find", modifiez le code de programme suivant :

```
...
MasterCopySystemFolder systemFolder = projectLibrary.MasterCopyFolder;
MasterCopyComposition mastercopies = systemFolder.MasterCopies;
MasterCopy masterCopy = mastercopies.Find("Copy of ...");
...
```

Pour énumérer les groupes et sous-groupes des copies maîtres, modifiez le code de programme suivant :

```
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)
    {
        ...
    }
}
```

Pour accéder à un MasterCopyUserFolder par la méthode "Find", modifiez le code de programme suivant :

```
...
MasterCopyUserFolderComposition userFolderComposition = ...
MasterCopyUserFolder userFolder = userFolderComposition.Find("Name of user folder");
...
```

Pour renommer une copie maître, modifiez le code de programme suivant :

```
//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")});
```

Interroger les informations des copies maîtres

Pour appeler les informations d'une copie maître, modifiez le code de programme suivant :

```
public static void GetMasterCopyInformation(MasterCopy masterCopy)
{
    string author = masterCopy.Author;
    DateTime creationDate = masterCopy.CreationDate;
    string name = masterCopy.Name;
}
```

Interroger les informations des objets dans une copie maître

L'objet MasterCopy contient le navigateur ContentDescriptions qui est une composition de MasterCopyContentDescriptions.

Une copie maître peut contenir plusieurs objets. L'objet MasterCopy contient des ContentDescriptions pour chaque objet existant directement dans la copie maître. Lorsque la copie maître comporte un dossier qui contient également quelques éléments, l'objet MasterCopy ne contient qu'une ContentDescription du dossier.

```
MasterCopy multiObjectMasterCopy = ...;

//Using ContentDescriptions
MasterCopyContentDescriptionComposition masterCopyContentDescriptions =
multiObjectMasterCopy.ContentDescriptions;
MasterCopyContentDescription contentDescription= masterCopyContentDescriptions.First();

string name = contentDescription.ContentName;
Type type = contentDescription.ContentType;
```

7.12.12 Créer la copie maîtresse d'un projet dans la bibliothèque

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Lorsqu'il s'agit d'une bibliothèque en lecture/écriture, vous pouvez créer une copie maître issue d'une `IMasterCopySource` sur l'emplacement cible.

```
MasterCopy  
MasterCopyComposition.Create(Siemens.Engineering.Library.MasterCopies.IMasterCopySource  
sourceObject);
```

Une exception `EngineeringException` est déclenchée dans les cas suivants :

- L'emplacement cible est accessible en lecture seule
- Le système refuse la génération d'une copie maître issue de la source

Les éléments suivants sont définis comme `IMasterCopySources` :

- Device - HW
- DeviceItem - HW
- DeviceUserGroup - HW
- CodeBlock - SW
- DataBlock - SW
- PlcBlockUserGroup - SW
- PlcTag - SW
- PlcTagTable - SW
- PlcTagTableUserGroup - SW
- PlcType - SW
- PlcTypeUserGroup - SW
- VBScript - HMI
- VBScriptUserFolder - HMI
- Screen - HMI
- ScreenTemplate - HMI
- ScreenTemplateUserFolder - HMI
- ScreenUserFolder - HMI

- Tag - HMI
- TagTable - HMI
- TagUserFolder - HMI

Code du programme

Utilisez le code de programme suivant :

```
// 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);
}
```

7.12.13 Créer un objet à partir d'une copie maîtresse

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Utilisation

L'interface TIA Portal Openness API prend en charge l'utilisation de copies maîtres dans le projet. Avec la méthode `CreateFrom`, vous pouvez créer un objet dans la composition de l'objet issu d'une copie maître dans une bibliothèque du projet ou une bibliothèque globale.

Le type fourni correspond à celui de la composition respective.

La méthode `CreateFrom` ne prend en charge que les copies maîtres contenant des objets isolés. Quand la composition dans laquelle l'action est appelée et la copie maître source sont incompatibles (par ex. quand la copie maître source contient une table de variables API et que la composition est celle d'un bloc API), une exception récupérable est lancée.

Les compositions suivantes sont prises en charge :

- `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

Code de programme : créer un bloc API à partir d'une copie maître

Pour créer un bloc API à partir d'une copie maître dans une bibliothèque, modifiez le code de programme suivant :

```
var plcSoftware = ...;
MasterCopy copyOfPlcBlock = ...;
PlcBlock plcSoftware.BlockGroup.Blocks.CreateFrom(copyOfPlcBlock);
```

Code de programme : créer un appareil à partir d'une copie maître

Pour créer un appareil à partir d'une copie maître dans une bibliothèque, modifiez le code de programme suivant :

```
Project project = ...;
MasterCopy copyOfDevice = ...;
Device newDevice = project.Devices.CreateFrom(copyOfDevice);
```

Code de programme : créer un élément d'appareil à partir d'une copie maître

Pour créer un élément d'appareil à partir d'une copie maître dans une bibliothèque, modifiez le code de programme suivant :

```
Device device = ...;
MasterCopy copyOfDeviceItem = ...;
DeviceItem newDeviceItem = device.DeviceItems.CreateFrom(copyOfDeviceItem);
```

Code de programme : créer un sous-réseau à partir d'une copie maître

Pour créer un sous-réseau à partir d'une copie maître dans une bibliothèque, modifiez le code de programme suivant :

```
Project project = ...;
MasterCopy copyOfSubnet = ...;
Subnet newSubnet = project.Subnets.CreateFrom(copyOfSubnet);
```

Code de programme : créer un dossier d'appareils à partir d'une copie maître

Pour créer un dossier d'appareils à partir d'une copie maître dans une bibliothèque, modifiez le code de programme suivant :

```
Project project = ...;
MasterCopy copyOfDeviceGroup = ...;
DeviceGroup newDeviceGroup= project.DeviceGroups.CreateFrom(copyOfDeviceGroup);
```

Voir aussi

Accéder à des modèles de copie (Page 144)

7.12.14 Copier des copies maîtresse

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'interface TIA Portal Openness API prend en charge la copie de copies maître au sein d'une même bibliothèque et d'une bibliothèque vers une autre à l'aide de l'action CreateFrom. L'action crée un nouvel objet sur la base de la copie maître source et le place dans la composition là où l'action a été appelée. L'action essaie de créer la nouvelle copie maître sous un nom identique à celui de la copie maître source. Lorsque ce nom n'est pas disponible, le système attribue un nouveau nom à la nouvelle copie maître. Ensuite, il affiche la copie maître.

Lorsque la composition dans laquelle l'action "CreateFrom" est appelée se trouve dans une bibliothèque globale protégée en écriture, une exception récupérable est déclenchée.

Code de programme

Modifiez le code de programme suivant :

```
ProjectLibrary projectLibrary = ...;

MasterCopy copiedMasterCopy =
projectLibrary.MasterCopyFolder.MasterCopies.CreateFrom(sampleMasterCopy)
```

Voir aussi

Accéder à des modèles de copie (Page 144)

7.12.15 Déterminer les instances de type obsolètes

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Vous avez accès à la bibliothèque requise.
Voir Accès aux bibliothèques globales (Page 126)
- Vous avez accès à un dossier pour les types.
Voir Accéder aux dossiers dans une bibliothèque (Page 132).

Utilisation

L'interface TIA Portal OpennessAPI prend en charge la détermination des versions de types faisant partie des instances dans le projet ouvert. L'interface TIA Portal OpennessAPI fournit l'un des deux états suivants pour chaque instance :

- L'instance se rapporte à une version de type obsolète.
- L'instance se rapporte à la version de type actuelle.

Les règles applicables pour la détermination de version sont les suivantes :

- La détermination de version est basée sur une bibliothèque et le projet que vous souhaitez ouvrir via l'interface TIA Portal OpennessAPI.
- Aucune instance n'est actualisée dans le cadre de la détermination de version.

Signature

Vous pouvez déterminer les instances d'une version de type par la méthode `UpdateCheck` :
`UpdateCheck(Project project, UpdateCheckMode updateCheckMode)`

Paramètres	Fonction
Project	Définit le projet dans lequel les versions de types des instances sont déterminées.
UpdateCheckMode	Indique les versions qui sont déterminées : <ul style="list-style-type: none"> ReportOutOfDateOnly : fournit en retour seulement l'état de type "obsolète". ReportOutOfDateAndUpToDate : fournit en retour l'état des types "obsolète" et "actuel" :

Résultat

Lors de la détermination de version, les appareils du projet sont interrogés de haut en bas. La présence d'une instance d'une version de type issue de la bibliothèque indiquée dans les données de configuration de l'appareil est contrôlée pour chaque appareil. La méthode `UpdateCheck` fournit le résultat de la vérification de version selon un ordre hiérarchique.

Le tableau ci-après montre le résultat d'une vérification de version avec le paramètre `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
	\HMI_1\Screens Screen_10 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

Code du programme

Il n'existe aucune différence entre les bibliothèques de projet et les bibliothèques globales concernant la manipulation du contrôle de la mise jour.

Pour déterminer les versions de type d'une bibliothèque globale ou d'une bibliothèque de projet pour les instances dans le projet, modifiez le code de programme suivant :

```
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);
}
```

Pour afficher le résultat de la détermination de version et parcourir les messages un par un, modifiez le code de programme suivant :

```
private static void RecursivelyWriteMessages (UpdateCheckResultMessageComposition
messages, string indent = "")
{
    indent += "\t";
    foreach (UpdateCheckResultMessage message in messages)
    {
        Console.WriteLine(indent + message.Description);
        RecursivelyWriteMessages(message.Messages, indent);
    }
}
```

Pour accéder à certaines parties de message dans le résultat de la détermination de version, modifiez le code de programme suivant :

```
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);
    }
}
```

7.12.16 Actualiser un projet

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)
- Vous avez accès à la bibliothèque requise.
Voir Accès aux bibliothèques globales (Page 126).
- Vous avez accès à un dossier pour types.
Voir Accéder aux dossiers dans une bibliothèque (Page 132).

Utilisation

L'interface TIA Portal Openness API vous permet de mettre à jour des instances d'un type sélectionné dans un dossier de types d'un projet.

Lors de l'actualisation, les instances utilisées dans le projet sont actualisées sur la base de la dernière version de type validée. Si vous actualisez les instances d'une bibliothèque globale, une synchronisation est d'abord exécutée.

Signature

Utilisez la méthode `UpdateProject` pour actualiser des instances.

Pour les classes implémentant l'interface `LibraryTypes`, utilisez l'appel suivant :

```
void UpdateProject(IUpdateProjectScope updateProjectScope)
```

Pour les classes implémentant l'interface `ILibrary`, utilisez l'appel suivant :

```
void UpdateProject(IEnumerable<ILibraryTypeOrFolderSelection>
selectedTypesOrFolders, IEnumerable <IUpdateProjectScope>
updateProjectScope)
```

Chaque appel est entré dans le fichier-journal du répertoire de projet.

Paramètre	Fonction
<code>IEnumerable<ILibraryTypeOrFolderSelection> selectedTypesOrFolders</code>	Indique les dossiers ou types devant être synchronisés ou dont les instances doivent être actualisées dans le projet.
<code>IUpdateProjectScope updateProjectScope</code> <code>IEnumerable <IUpdateProjectScope> updateProjectScope</code>	Indique dans le projet les objets dans lesquels les utilisations des instances sont actualisées. Les objets suivants sont pris en charge : <ul style="list-style-type: none"> • <code>PlcSoftware</code> • <code>HmiTarget</code>

Code de programme

Pour actualiser les instances de types sélectionnés dans un dossier de types, modifiez le code de programme suivant :

```
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);
}
```

7.12.17 Actualiser une bibliothèque

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)
- Vous avez accès à la bibliothèque requise.
Voir Accès aux bibliothèques globales (Page 126).
- Vous avez accès à un dossier pour types.
Voir Accéder aux dossiers dans une bibliothèque (Page 132).

Utilisation

L'interface TIA Portal Openness API prend en charge les actualisations suivantes dans la bibliothèque de projet :

- Synchronisation des types sélectionnés entre les bibliothèques.

La structure des dossiers n'est pas adaptée lors de la synchronisation. Les types à actualiser sont déterminés et actualisés à l'aide de leur GUID :

- Si un type d'une bibliothèque comporte une version de type manquant dans la bibliothèque devant être actualisée, la version de type est copiée.
- L'opération est annulée et une Exception est émise, si un type d'une bibliothèque comporte une version de type avec différents GUID.

Signature

Pour synchroniser les versions de type, utilisez la méthode `UpdateLibrary`.

Pour les classes implémentant l'interface `LibraryTypes`, utilisez l'appel suivant :

```
void UpdateLibrary(ILibrary targetLibrary)
```

Pour les classes implémentant l'interface `ILibrary`, utilisez l'appel suivant :

```
void UpdateLibrary(IEnumerable<LibraryTypeOrFolderSelection> selectedTypesOrFolders, ILibrary targetLibrary)
```

Paramètre	Fonction
<code>IEnumerable<ILibraryTypeOrFolderSelection> selectedTypesOrFolders</code>	Indique les dossiers ou types devant être synchronisés ou dont les instances doivent être actualisées dans le projet.
<code>ILibrary targetLibrary</code>	Indique la bibliothèque dont le contenu doit être synchronisé avec une bibliothèque. Si la bibliothèque source et la bibliothèque cible sont identiques, une exception est lancée.

Code de programme

Pour synchroniser un type d'une bibliothèque de projet avec une bibliothèque globale, modifiez le code de programme suivant :

```
sourceType.UpdateLibrary(projectLibrary);
```

Pour synchroniser des types sélectionnés dans un dossier de types entre une bibliothèque globale et la bibliothèque de projet, modifiez le code de programme suivant :

```
globalLibrary.UpdateLibrary(new[]{globalLibrary.TypeFolder}, projectLibrary);
```

7.12.18 Supprimer les contenus de bibliothèque

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)
- Vous avez accès à la bibliothèque requise.
Voir Accès aux bibliothèques globales (Page 126).
- Vous avez accès à un dossier pour les types.
Voir Accéder aux dossiers dans une bibliothèque (Page 132).

Utilisation

L'interface TIA Portal Openness API vous permet de supprimer les contenus suivants dans la bibliothèque du projet :

- Types
- Version de type
- Dossiers personnalisés pour types
- Copies maître
- Dossiers personnalisés pour copies maître

Remarque

Suppression de types et de dossiers avec des types personnalisés

Si vous souhaitez supprimer un type ou un dossier avec des types personnalisés, les "Règles de suppression de versions" doivent être respectées. Vous pouvez supprimer un dossier de type vide à tout moment.

Remarque**Règles de suppression de versions**

Seules les versions au statut "Committed" peuvent être supprimées. Pour la suppression de versions, les règles à respecter sont les suivantes :

- Si vous venez de créer une version au statut "InWork" depuis une version au statut "Committed", vous ne pouvez supprimer la version au statut "Committed" que lorsque la nouvelle version est annulée ou obtient le statut "Committed".
 - Si un type ne possède qu'une seule version, le type est également supprimé.
 - Si la version A dépend de la version B d'un autre type, supprimez d'abord la version A, puis la version B.
 - Si des instances de la version A existent, vous ne pouvez supprimer la version A que si vous supprimez également les instances. De plus, si une instance se trouve dans une copie maître, celle-ci est également supprimée.
-

Code du programme

Pour supprimer des types ou des dossiers avec des types personnalisés, modifiez le code de programme suivant :

```
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();
}
```

Pour supprimer un type ou dossier avec des types personnalisés en particulier, modifiez le code de programme suivant :

```
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();
}
```

Pour supprimer une version, modifiez le code de programme suivant :

```
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();
}
```

Pour supprimer une copie maître ou un dossier avec des copies maîtres personnalisées, modifiez le code de programme suivant :

```
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();
}
```

Voir aussi

[Accéder à des modèles de copie \(Page 144\)](#)

7.13 Fonctions pour l'appel d'appareils, de réseaux et de liaisons

7.13.1 Ouvrir l'éditeur "Appareils & réseaux"

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Vous pouvez ouvrir l'éditeur "Appareils & réseaux" avec l'interface API grâce à l'une des deux méthodes suivantes :

- `ShowHwEditor(View.Topology ou View.Network ou View.Device)` : Ouvre l'éditeur "Appareils & Réseaux" depuis le projet.
- `ShowInEditor(View.Topology ou View.Network ou View.Device)` : Affiche l'appareil indiqué dans l'éditeur "Appareils & Réseaux".

Le paramètre `View` vous permet de définir la vue qui est affichée lors de l'ouverture de l'éditeur :

- `View.Topology`
- `View.Network`
- `View.Device`

Code du programme

Pour ouvrir l'éditeur "Appareils & Réseaux", modifiez le code de programme suivant :

```
// Open topology view from project
private static void OpenEditorDevicesAndNetworksFromProject(Project project)
{
    project.ShowHwEditor(Siemens.Engineering.HW.View.Topology);
}
```

Pour ouvrir l'éditeur "Appareils & Réseaux" pour un appareil, modifiez le code de programme suivant :

```
// Open topology view for given device
private static void OpenEditorDevicesAndNetworksFromDevice(Device device)
{
    device.ShowInEditor(Siemens.Engineering.HW.View.Topology);
}
```


Voir aussi

Importation de données de configuration (Page 377)

7.13.2 Interroger PLC Target et HMI Target

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Vous pouvez définir si un logiciel de base peut être utilisé comme PLC Target (PlcSoftware) ou HMI Target dans la TIA Portal Openness API.

Code du programme : PLC Target

Pour vérifier si un élément d'appareil peut être utilisé comme PLC Target, utilisez le code de programme suivant :

```
// 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;
}
```

Code du programme : HMI Target

Pour vérifier si un élément d'appareil peut être utilisé comme HMI Target, modifiez le code de programme suivant :

```
//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;
}
```

Voir aussi

Énumérer des appareils (Page 211)

7.13.3 Accéder à des attributs d'un objet adresse

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne pour l'accès en écriture.

Utilisation

L'interface TIA Portal Openness API vous permet d'appeler ou de déterminer des attributs de l'objet adresse.

En outre, vous pouvez affecter la mémoire image actuelle à un OB.

Il est possible d'accéder aux attributs suivants :

Nom d'attribut	Type de données	Acces-sible en écriture	Accès	Description
IsochronousMode	BOOL	r/w	Attribut dynamique	Activer/désactiver le mode isochrone
ProcessImage	Int32	r/w	Attribut dynamique	Déterminer/appeler le numéro de partition de la mémoire image
InterruptObNumber	Int64	r/w	Attribut dynamique	Déterminer/appeler le numéro du bloc d'organisation d'alarme (uniquement contrôleur classique)
StartAddress	Int32	r/w	Attribut modélisé	Déterminer/appeler nouvelle valeur pour StartAddress

Restrictions

- Attribut `StartAddress`
 - Déterminer `StartAddress` peut modifier implicitement la `StartAddress` du type d'E/S opposé dans le module nom. La modification de l'adresse d'entrée modifie l'adresse de sortie.
 - L'accès en écriture n'est pas pris en charge pour tous les appareils.
 - Les adresses comprimées ne sont pas prises en charge dans TIA Portal Openness.
 - Les variables affectées ne sont pas recâblées par une modification de l'adresse avec TIA Portal Openness.
- Attribut `InterruptObNumber`
 - Accessible uniquement dans les paramétrages avec contrôleurs S7-300 ou S7-400. L'accès en écriture est pris en charge pour les contrôleurs S7-400.

Code de programme : appeler ou déterminer des attributs d'un objet adresse

Pour accéder au mode isochrone d'un objet adresse, modifiez le code de programme suivant :

```
Address address= ...;

// read attribute
bool attributeValue = (bool)address.GetAttribute("IsochronousMode");

// write attribute
address.SetAttribute("IsochronousMode", true);
```

7.13 Fonctions pour l'appel d'appareils, de réseaux et de liaisons

Pour accéder à l'attribut `ProcessImage` d'un objet adresse, modifiez le code de programme suivant :

```
Address address= ...;

// read attribute
int attributeValue = (int)address.GetAttribute("ProcessImage");

// write attribute
address.SetAttribute("ProcessImage", 7);
```

Pour accéder à l'attribut `InterruptObNumber` d'un objet adresse, modifiez le code de programme suivant :

```
Address address= ...;

// read attribute
long attributeValue = (long)address.GetAttribute("InterruptObNumber");

// write attribute
address.SetAttribute("InterruptObNumber", 42L);

//default value = 40
```

Pour accéder à l'attribut `StartAddress` d'un objet adresse, modifiez le code de programme suivant :

```
Address address= ...;

// read attribute
int attributeValue = (int)address.GetAttribute("StartAddress");

// write attribute
address.StartAddress = IntValueStartAddress;
```

Code de programme : Affecter la mémoire image actuelle à un OB

Pour affecter la mémoire image actuelle à un OB, modifiez le code de programme suivant :

```
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);
}
```

7.13.4 Appeler une voie de module

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les modules d'entrées-sorties tels que les modules d'entrées analogiques comportent normalement plusieurs voies par module. Habituellement, ces voies offrent plusieurs fonctions simultanées, par exemple un module d'entrées analogiques à quatre voies peut mesurer quatre valeurs de tension en même temps.

Pour appeler toutes les voies d'un module, on utilise l'attribut de voie d'un élément d'appareil.

Code de programme : Attributs des voies

Pour accéder aux attributs d'une voie, modifiez le code de programme suivant :

```
DeviceItem aiModule = ...
ChannelComposition channels = aiModule.Channels;
foreach (Channel channel in channels)
{
    ... // Work with the channel
}
```

Code de programme : déterminer des attributs

Pour appeler les attributs d'identification pour les différentes voies, modifiez le code de programme suivant :

```
Channel channel = ...
int channelNumber = channel.Number;
ChannelType type = channel.Type;
ChannelIoType ioType = channel.IoType;
```

Code de programme : appeler une voie particulière

Pour utiliser les attributs d'identification pour l'accès direct à une voie, modifiez le code de programme suivant :

```
DeviceItem aiModule = ...
Channel channel = aiModule.Channels.Find(ChannelType.Analog, ChannelIoType.Input, 0);
... // Work with the channel
```

Types de voie

Valeur	Description
ChannelType.None	Type de voie non valable.
ChannelType.Analog	Type de voie analogique.
ChannelType.Digital	Type de voie TOR.
ChannelType.Technology	Type de voie technologique.

Types ES de voie

Valeur	Description
ChannelIoType.None	Type ES de voie non valable.
ChannelIoType.Input	Une voie d'entrée.
ChannelIoType.Output	Une voie de sortie.
ChannelIoType.Complex	Types ES complexes, par ex. pour voies technologiques.

7.13.5 Utilisation d'associations

Accéder aux affectations

Une affectation décrit la relation entre deux objets ou plus au niveau du type.

TIA Portal Openness prend en charge l'accès aux affectations via l'index et les boucles "foreach". L'accès direct, par exemple via string name, n'est pas pris en charge.

Attributs

Les attributs suivants sont disponibles :

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

Méthodes

TIA Portal Openness prend en charge les méthodes suivantes :

- `int IndexOf (type)` : fournit en retour l'index dans l'affectation pour une instance transmise :
- `bool Contains (type)` : détermine si l'instance transmise est contenue dans l'affectation.
- `IEnumerator GetEnumerator <retType> ()` : est utilisé dans des boucles foreach et permet d'accéder à un objet.
- `void Add (type)1` : ajoute l'instance transmise de l'affectation.
- `void Remove (type)1` : supprime l'instance transmise de l'affectation.

¹ : n'est pas prise en charge par toutes les affectations.

7.13.6 Utilisation de compositions

Appel de compositions

Une composition est un cas particulier d'affectation. Une composition exprime une relation sémantique entre deux objets dont l'un est une partie de l'autre.

Attributs

Les attributs suivants sont disponibles :

- `int Count`
- `bool IsReadOnly`

- `IEngineeringObject Parent`
- `retType this [int index] {get;}` : accède de manière indexée à un objet de la composition.
Vous devez utiliser ce type d'accès de manière ciblée uniquement, car chaque accès indexé dépasse les limites du processus.

Méthodes

TIA Portal Openness prend en charge les méthodes suivantes :

- `retType Create (id, ...)` : crée une nouvelle instance et l'ajoute à la composition. La signature de la méthode dépend du type de création de l'instance. Cette méthode n'est pas prise en charge par toutes les compositions.
- `type Find (id, ...)` : Recherche dans une composition l'instance ayant l'ID transmis. La recherche n'est pas récursive. La signature de la méthode dépend du type de recherche de l'instance. Cette méthode n'est pas prise en charge par toutes les compositions.
- `IEnumerator GetEnumerator<retType> ()` : est utilisé dans des boucles foreach et permet d'accéder à un objet.
- `Delete (type)1` : supprime l'instance spécifiée par la référence d'objet actuelle.
- `int IndexOf (type)` : fournit en retour l'index dans la composition pour une instance transmise.
- `bool Contains (type)` : détermine si l'instance transmise est contenue dans la composition.
- `void Import(string path, ImportOptions importOptions)1` : S'applique à chaque composition comportant des types pouvant être importés. Chaque signature d'importation contient un paramètre de configuration du type "ImportOptions (Page 377)" ("None", "Overwrite") qui permet à l'utilisateur de commander le comportement à l'importation.

¹ : Pas prise en charge par toutes les compositions.

7.13.7 Vérifier l'égalité des objets

Utilisation

En tant qu'utilisateur d'une TIA Portal Openness API, vous pouvez vérifier la similitude des objets à l'aide du code de programme :

- Vous vérifiez alors avec l'opérateur `==` si deux références d'objet sont identiques.
- La méthode `System.Object.Equals()` vous permet de vérifier si deux objets sont réellement identiques en ce qui concerne le portail TIA.

Code du programme

Pour vérifier les types de référence d'objet, modifiez le code de programme suivant :

```
...
//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");
}
```

7.13.8 Opérations de lecture pour attributs

Opérations d'ensemble et opérations de lecture standards pour attributs

TIA Portal Openness prend en charge l'accès aux attributs par les méthodes suivantes, disponibles au niveau de l'objet :

- Opérations d'ensemble pour l'accès en lecture
- Opérations standard de lecture

Code du programme pour les opérations d'ensemble

```
//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]);
    }
}
```

Opérations d'ensemble pour l'accès en lecture

Cette méthode est disponible pour chaque objet :

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

Opérations standard de lecture

Les opérations suivantes sont disponibles :

- Appeler les noms des attributs disponibles :
Utilisez la méthode `GetAttributeInfos()` (Page 96) sur un `IEngineeringObject`.
- Méthode générique pour la lecture d'un attribut
`public abstract object GetAttribute(string name);`

Remarque

Les attributs dynamiques ne s'affichent pas dans IntelliSense car leur disponibilité dépend de l'état de l'instance d'objet.

7.14 Fonctions dans les réseaux

7.14.1 Créer un sous-réseau

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Il y a deux manières différentes de créer des sous-réseaux :

- Création d'un sous-réseau qui est relié à une interface : le type de l'interface pour laquelle le sous-réseau est créé détermine alors le type du sous-réseau.
- Création d'un sous-réseau qui n'est pas relié à une interface.

Code de programme : créer un sous-réseau qui est relié à une interface

Pour créer un sous-réseau, modifiez le code de programme suivant :

```
Node node = ...;  
Subnet subnet = node.CreateAndConnectToSubnet("NameOfSubnet");
```

Les identifiants de type suivants sont utilisés :

- System:Subnet.Ethernet
- System:Subnet.Profibus
- System:Subnet.Mpi
- System:Subnet.Asi

Code de programme : créer un sous-réseau qui n'est pas relié à une interface

Pour créer un sous-réseau, modifiez le code de programme suivant :

```
Project project = ...;  
SubnetComposition subnets = project.Subnets;  
  
Subnet newSubnet = subnets.Create("System:Subnet.Ethernet", "NewSubnet");
```

Les identifiants de type suivants peuvent être utilisés :

- System:Subnet.Ethernet
- System:Subnet.Profibus
- System:Subnet.Mpi
- System:Subnet.Asi

7.14.2 Accéder aux sous-réseaux

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Pour plusieurs caractéristiques relatives aux sous-réseaux, par ex. pour leur affecter des interfaces, vous devez appeler des sous-réseaux dans le projet. Normalement, les sous-réseaux sont réunis directement au niveau du projet.

Code de programme : accéder à tous les sous-réseaux d'un projet

Pour accéder à tous les sous-réseaux à l'exception des sous-réseaux internes d'un projet, modifiez le code de programme suivant :

```
Project project = ...
foreach (Subnet net in project.Subnets)
{
    ... // Work with the subnet
}
```

Code de programme : accéder à un sous-réseau spécifique

Pour accéder à un sous-réseau précis grâce à son nom, modifiez le code de programme suivant :

```
Project project = ...
Subnet net = project.Subnets.Find("PROFIBUS_1");
{
    ... // Work with the subnet
}
```

Attributs d'un sous-réseau

Un sous-réseau possède les attributs suivants :

```
Subnet net = ...;
string name = net.Name;
NetType type = net.NetType;
```

Types de réseau

Valeur	Description
NetType.Unknown	Le type du réseau est inconnu.
NetType.Ethernet	Le type du réseau est Ethernet.
NetType.Profibus	Le type du réseau est Profibus.
NetType.Mpi	Le type du réseau est MPI.
NetType.ProfibusIntegrated	Le type du réseau est Profibus intégré.
NetType.Asi	Le type du réseau est ASi.
NetType.PcInternal	Le type du réseau est PC interne.
NetType.Ptp	Le type du réseau est PtP.
NetType.Link	Le type du réseau est Link.
NetType.Wan	Le type du réseau est Wide Area Network.

7.14.3 Accéder à des sous-réseaux internes

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Quand un élément d'appareil peut former un sous-réseau, il dispose de la fonction supplémentaire "Propriétaire de sous-réseau". Pour accéder à cette fonction supplémentaire, il faut utiliser un certain service de l'élément d'appareil.

Code de programme : appeler le rôle de propriétaire de sous-réseau

Pour appeler le rôle de propriétaire de sous-réseau, modifiez le code de programme suivant :

```
SubnetOwner subnetOwner =
((IEngineeringServiceProvider) deviceItem).GetService<SubnetOwner>();
if (subnetOwner != null)
{
    // work with the role
}
```

Code de programme : Attributs d'un propriétaire de sous-réseau

Pour accéder aux sous-réseaux d'un propriétaire de sous-réseau, utilisez le code de programme suivant :

```
foreach(Subnet subnet in subnetOwner.Subnets)
{
    Subnet interalSubnet = subnet;
}
```

7.14.4 Appeler l'identifiant de type de sous-réseaux**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'attribut `TypeIdentifier` est utilisé pour identifier un sous-réseau. `TypeIdentifier` est une chaîne de caractères composée de plusieurs parties :

```
<TypeIdentifierType>:<SystemIdentifier>
```

Les valeurs possibles pour `TypeIdentifierType` sont :

- System

SystemIdentifier

Type de sous-réseau	Identifiant du système
PROFIBUS	Subnet.Profibus
MPI	Subnet.Mpi
Industrial Ethernet	Subnet.Ethernet

Type de sous-réseau	Identifiant du système
ASI	Subnet.Asi
PtP	Subnet.Ptp
PROFIBUS - intégré	Subnet.ProfibusIntegrated
PC - interne	<i>null</i>

Code de programme

Pour appeler l'identifiant de type des objets pour GSD que l'utilisateur peut créer séparément et gérer, modifiez le code de programme suivant :

```
Subnet subnet = ...;
string typeIdentifieur = subnet.TypeIdentifieur;
```

7.14.5 Appeler les attributs d'un sous-réseau

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Un sous-réseau possède certains attributs obligatoires pouvant être lus et/ou écrits. Les attributs ne sont disponibles que s'ils sont visibles dans l'interface utilisateur. En règle générale, l'opération d'écriture n'est autorisée que sur un attribut que l'utilisateur peut également modifier dans l'UI. Cela peut varier en fonction du type de sous-réseau. L'utilisateur ne peut définir DpCycleTime que si IsochronousMode est vrai et DpCycleMinTimeAutoCalculation est faux.

Attributs de sous-réseaux de type ASI

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r/w		Nom du sous-réseau.
NetType	NetType	r		Type du sous-réseau.
SubnetId	String	r	Dyna- mique	ID univoque du sous-réseau. L'ID de sous-réseau S7 est composée de deux nombres reliés par un trait d'union. Un nombre pour le projet et un pour le sous-réseau, par exemple 4493-1.

Attributs de sous-réseaux de type Ethernet

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r/w		Nom du sous-réseau.
NetType	NetType	r		Type du sous-réseau.
SubnetId	String	r/w	Dyna- mique	ID univoque du sous-réseau. L'ID de sous-réseau S7 est composée de deux nombres reliés par un trait d'union. Un nombre pour le projet et un pour le sous-réseau, par exemple 4493-1.
DefaultSubnet	Bool	r/w	Dyna- mique	Vrai s'il s'agit d'un sous-réseau par défaut. Dans un projet, il existe un sous-réseau par défaut au maximum.

Attributs de sous-réseaux de type MPI

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r/w		Nom du sous-réseau.
NetType	NetType	r		Type du sous-réseau.
SubnetId	String	r/w	Dyna- mique	ID univoque du sous-réseau. L'ID de sous-réseau S7 est composée de deux nombres reliés par un trait d'union. Un nombre pour le projet et un pour le sous-réseau, par exemple 4493-1.
HighestAddress	Int	r/w	Dyna- mique	Adresse MPI la plus élevée du sous-réseau.
TransmissionSpeed	BaudRate	r/w	Dyna- mique	Vrai s'il s'agit d'un sous-réseau par défaut. Dans un projet, il existe un sous-réseau par défaut au maximum.

Attributs de sous-réseaux de type PC internal

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r		Nom du sous-réseau.
NetType	NetType	r		Type du sous-réseau.
SubnetId	String	r	Dyna- mique	ID univoque du sous-réseau. L'ID de sous-réseau S7 est composée de deux nombres reliés par un trait d'union. Un nombre pour le projet et un pour le sous-réseau, par exemple 4493-1.

Attributs de sous-réseaux de type PROFIBUS

Attribut	Type de données	Acces-sible en écriture	Accès	Description
Name	String	r/w		Nom du sous-réseau.
NetType	NetType	r		Type du sous-réseau.
SubnetId	String	r/w	Dyna-mique	ID univoque du sous-réseau. L'ID de sous-réseau S7 est composée de deux nombres reliés par un trait d'union. Un nombre pour le projet et un pour le sous-réseau, par exemple 4493-1.
HighestAddress	Int	r/w	Dyna-mique	Adresse PROFIBUS la plus élevée du sous-réseau.
TransmissionSpeed	BaudRate	r/w	Dyna-mique	Vrai s'il s'agit d'un sous-réseau par défaut. Dans un projet, il existe un sous-réseau par défaut au maximum.
BusProfile	Profil de bus	r/w	Dyna-mique	Le profil PROFIBUS.
PbCableConfiguration	Bool	r/w	Dyna-mique	Vrai, pour activer des paramètres réseau PROFIBUS supplémentaires
PbRepeaterCount	Int	r/w	Dyna-mique	Nombre de répéteurs pour conducteur en cuivre
PbCopperCableLength	double	r/w	Dyna-mique	La longueur du conducteur en cuivre
PbOpticalComponentCount	Int	r/w	Dyna-mique	Nombre de OLM et OBT du câble en fibre optique.
PbOpticalCableLength	double	r/w	Dyna-mique	La longueur en km du câble en fibre optique pour le réseau PROFIBUS.
PbOpticalRing	Bool	r/w	Dyna-mique	Vrai si des paramètres du bus sont repris pour un anneau optique.
PbOlmP12	Bool	r/w	Dyna-mique	Vrai si OLM/P12 est activé pour le calcul du paramètre du bus.
PbOlmG12	Bool	r/w	Dyna-mique	Vrai si OLM/G12 est activé pour le calcul du paramètre du bus.
PbOlmG12Eec	Bool	r/w	Dyna-mique	Vrai si OLM/G12-EEC est activé pour le calcul du paramètre du bus.
PbOlmG121300	Bool	r/w	Dyna-mique	Vrai si OLM/G12-1300 est activé pour le calcul du paramètre du bus.
PbAdditionalNetworkDevices	Bool	r/w	Dyna-mique	Vrai, si des appareils de bus supplémentaires existant dans le projet sont pris en compte pour le calcul des cycles de bus.
PbAdditionalDpMaster	Int	r/w	Dyna-mique	Nombre de maîtres DP non configurés.
PbTotalDpMaster	Int	r	Dyna-mique	Nombre total de maîtres DP
PbAdditionalPassiveDevice	Int	r/w	Dyna-mique	Nombre d'esclaves DP non configurés ou d'appareils passifs.

Attribut	Type de données	Acces-sible en écriture	Accès	Description
PbTotalPassiveDevice	Int	r	Dyna-mique	Nombre total d'esclaves DP ou d'appareils passifs.
PbAdditionalActiveDevice	Int	r/w	Dyna-mique	Nombre d'appareils actifs non configurés avec charge de communication FDL/FMS/S/.
PbTotalActiveDevice	Int	r	Dyna-mique	Nombre total d'appareils actifs avec charge de communication FDL/FMS/S/.
PbAdditionalCommunicationLoad	CommunicationLoad	r/w	Dyna-mique	Estimation approximative de la charge de communication
PbDirectDateExchange	Bool	r/w	Dyna-mique	Optimisation pour échange direct de données.
PbMinimizeTslotForSlaveFailure	Bool	r/w	Dyna-mique	Réduction du temps d'affectation pour erreurs d'esclave.
PbOptimizeCableConfiguration	Bool	r/w	Dyna-mique	Optimisation de la configuration des câbles.
PbCyclicDistribution	Bool	r/w	Dyna-mique	Vrai, si le partage cyclique des paramètres de bus est activé.
PbTslotInit	Int	r/w	Dyna-mique	Valeur par défaut de Tslot.
PbTslot	Int	r	Dyna-mique	Temps d'attente de réception (slot time)
PbMinTsdr	Int	r/w	Dyna-mique	Temps minimal d'exécution du protocole
PbMaxTsdr	Int	r/w	Dyna-mique	Temps maximal d'exécution du protocole
PbTid1	Int	r	Dyna-mique	Temps de repos 1
PbTid2	Int	r	Dyna-mique	Temps de repos 2
PbTrdy	Int	r	Dyna-mique	Temps de disponibilité
PbTset	Int	r/w	Dyna-mique	Temps de configuration
PbTqui	Int	r/w	Dyna-mique	Temps d'atténuation du modulateur
PbTtr	int64	r/w	Dyna-mique	La valeur Ttr dans t_Bit.
PbTtrTypical	int64	r	Dyna-mique	Temps de réponse moyen sur le bus
PbWatchdog	int64	r/w	Dyna-mique	Surveillance de réponse
PbGapFactor	Int	r/w	Dyna-mique	Facteur d'actualisation GAP
PbRetryLimit	Int	r/w	Dyna-mique	Nombre maximum de tentatives

Attribut	Type de données	Acces-sible en écriture	Accès	Description
IsochronousMode	Bool	r/w	Dyna-mique	Vrai, si le temps de cycle de bus constant est activé.
PbAdditionalPassivDeviceForIsochronousMode	Int	r/w	Dyna-mique	Nombre de OP/PG/TD, etc. supplémentaires non configurés dans cette vue de réseau.
PbTotalPassivDeviceForIsochronousMode	Int	r	Dyna-mique	Somme d'appareils configurés et non configurés, par exemple OP/PG/TD, etc.
DpCycleMinTimeAutoCalculation	Bool	r/w	Dyna-mique	Vrai, si le calcul automatique et le réglage du temps de cycle DP le plus court sont activés.
DpCycleTime	double	r/w	Dyna-mique	Le temps de cycle DP.
IsochronousTiToAutoCalculation	Bool	r/w	Dyna-mique	Vrai, si le calcul automatique et le réglage des valeurs IsochronousTi et IsochronousTo sont activés.
IsochronousTi	double	r/w	Dyna-mique	Time Ti (temps de lecture des valeurs de process)
IsochronousTo	double	r/w	Dyna-mique	Time To (temps de sortie des valeurs de process)

Attributs de sous-réseaux de type PROFIBUS Integrated

Attribut	Type de données	Acces-sible en écriture	Accès	Description
Name	String	r/w		Nom du sous-réseau.
NetType	NetType	r		Type du sous-réseau.
SubnetId	String	r/w	Dyna-mique	ID univoque du sous-réseau. L'ID de sous-réseau S7 est composée de deux nombres reliés par un trait d'union. Un nombre pour le projet et un pour le sous-réseau, par exemple 4493-1.
IsochronousMode	Bool	r	Dyna-mique	Temps de cycle de bus constant activé.
DpCycleMinTimeAutoCalculation	Bool	r/w	Dyna-mique	Vrai, si le calcul automatique et le réglage du temps de cycle DP le plus court sont activés.
DpCycleTime	double	r/w	Dyna-mique	Le temps de cycle DP.
IsochronousTiToAutoCalculation	Bool	r/w	Dyna-mique	Vrai, si le calcul automatique et le réglage des valeurs IsochronousTi et IsochronousTo sont activés.
IsochronousTi	double	r/w	Dyna-mique	Time Ti (temps de lecture des valeurs de process)
IsochronousTo	double	r/w	Dyna-mique	Time To (temps de sortie des valeurs de process)

Code de programme

Pour appeler ou définir les attributs d'un sous-réseau, modifiez le code de programme suivant :

```
Subnet subnet = ...;

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");
```

Débit en bauds

Valeur	Description
BaudRate.None	Le débit en bauds est inconnu.
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 bauds
BaudRate.Baud1500000	1,5 MBaud
BaudRate.Baud3000000	3 MBaud
BaudRate.Baud6000000	6 MBaud
BaudRate.Baud12000000	12 MBaud

Profils de bus

Valeur	Description
BusProfile.None	Le profil du bus est inconnu.
BusProfile.DP	Le type du réseau est DP.
BusProfile.Standard	Le type du réseau est Standard.
BusProfile.Universal	Le type du réseau est Universal.
BusProfile.UserDefined	Le type du réseau est personnalisé.

Charge due à la communication

Valeur	Description
CommunicationLoad.None	Aucune charge de communication valide.
CommunicationLoad.Low	Typique pour la DP, pas de communication de données importante en dehors de la DP.

Valeur	Description
CommunicationLoad.Medium	Typique pour l'exploitation mixte de la DP et d'autres services de communication (p. ex. communication S7), si les exigences de la DP en matière de temps sont élevées et que le volume de communication acyclique est moyen.
CommunicationLoad.High	Pour l'exploitation mixte de la DP et d'autres services de communication (p. ex. communication S7), si les exigences de la DP en matière de temps sont faibles et que le volume de communication acyclique est élevé.

7.14.6 Supprimer un sous-réseau global

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code de programme

Pour supprimer un sous-réseau global d'un projet, modifiez le code de programme suivant :

```
Project project = ...;
SubnetComposition subnets = projects.Subnets;

// delete subnet
Subnet subnetToDelete = ...;
subnetToDelete.Delete();
```

7.14.7 Énumérer tous les abonnés d'un sous-réseau

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Énumération de tous les abonnés d'un sous-réseau

Code de programme

Pour énumérer les réseaux maître DP du sous-réseau, modifiez le code de programme suivant :

```
Subnet subnet = ...;
foreach (Node node in subnet.Nodes)
{
    // work with the node
}
```

7.14.8 Énumérer les réseaux IO d'un sous-réseau

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'énumération de IoSystem fournit tous les réseaux IO qui se trouvent dans le sous-réseau. Le système maître et le réseau IO sont représentés tous deux par la classe IoSystem.

Code de programme

Pour énumérer les réseaux maître DP du sous-réseau, modifiez le code de programme suivant :

```
Subnet subnet = ...;
foreach (IoSystem ioSystem in subnet.IoSystems)
{
    // work with the io system
}
```

7.14.9 Accéder aux abonnés

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'interface de rôle agrège les abonnés : Pour accéder aux attributs qui sont liés à l'affectation d'adresse et de sous-réseau d'une interface.

Le nom d'un abonné s'affiche dans les attributs d'une interface dans TIA Portal. L'identifiant d'abonné est un identifiant unique pour chaque abonné répertorié sur une interface. Cette valeur ne peut s'afficher qu'avec TIA Portal Openness.

Code de programme

Pour accéder à tous les abonnés d'une interface, modifiez le code de programme suivant :

```
NetworkInterface itf = ...
foreach (Node node in itf.Nodes)
{
    ... // Work with the node
}
```

La plupart des interfaces n'ont qu'un seul abonné, c'est pourquoi le premier abonné est utilisé normalement.

```
NetworkInterface itf = ...
Node node in itf.Nodes.First()
{
    ... // Work with the node
}
```

Les abonnés mettent à disposition leur nom, type et ID d'abonné en tant qu'attributs :

```
Node node = ...
string name = node.Name;
NetType type = node.NodeType;
string id = node.NodeId;
```

Types de réseau

Valeur	Description
NetType.Unknown	Le type du réseau est inconnu.
NetType.Ethernet	Le type du réseau est Ethernet.
NetType.Profibus	Le type du réseau est Profibus.
NetType.Mpi	Le type du réseau est MPI.
NetType.ProfibusIntegrated	Le type du réseau est Profibus intégré.
NetType.Asi	Le type du réseau est ASi.
NetType.PcInternal	Le type du réseau est PC interne.
NetType.Ptp	Le type du réseau est PtP.

7.14.10 Appeler les attributs d'un abonné

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Un élément d'appareil possède certains attributs obligatoires pouvant être lus et/ou écrits. Les attributs ne sont disponibles que s'ils sont visibles dans l'interface utilisateur. En règle générale, l'opération d'écriture n'est autorisée que sur un attribut que l'utilisateur peut également modifier dans l'UI. Cela peut varier en fonction du type d'élément d'appareil. L'utilisateur ne peut définir RouterAddress que si RouterUsed est vrai. Si l'utilisateur modifie le SubnetMask sur le contrôleur IO, le masque de sous-réseau est également modifié à la même valeur sur tous les périphériques IO.

Attributs d'un abonné de type ASI

Attributs	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r		Nom de l'abonné.
NodeId	String	r		ID de l'abonné.
NodeType	NetType	r		Un abonné appelle son type du sous-réseau.
Address	String	r/w	Dyna- mique	Autres attributs pour des esclaves AS-i.

Attributs d'un abonné de type Ethernet

Attributs	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r		Nom de l'abonné.
Nodeld	String	r		ID de l'abonné.
NodeType	NetType	r/w ou r		Un abonné appelle son type du sous-réseau.
UseIsoProtocol	Bool	r/w	Dyna- mique	
MacAddress	String	r/w	Dyna- mique	Par exemple 01-80-C2-00-00-00
UseIpProtocol	Bool	r/w	Dyna- mique	Cette valeur peut être lue même si elle n'est pas visible sur la commande TIA IU correspondante.
IpProtocolSelection	Enum	r/w	Dyna- mique	
Address	String	r/w	Dyna- mique	Seul IPv4 est pris en charge, pas IPv6
SubnetMask	String	r/w	Dyna- mique	
UseRouter	Bool	r/w	Dyna- mique	
RouterAddress	String	r/w	Dyna- mique	
DhcpClientId	String	r/w	Dyna- mique	
PnDeviceNameSetDirectly	Bool	r/w	Dyna- mique	Le nom d'appareil PROFINET est défini directement sur l'appareil. Pas disponible sur chaque appareil.
PnDeviceNameAutoGene- ration	Bool	r/w	Dyna- mique	Le nom d'appareil PROFINET est créé automatique- ment.
PnDeviceName	String	r/w	Dyna- mique	Nom univoque dans le sous-réseau.
PnDeviceNameConverted	String	r	Dyna- mique	Le nom d'appareil est converti pour une utilisation en interne par le système.

Attributs d'un abonné de type MPI

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r		Nom de l'abonné.
Nodeld	String	r		ID de l'abonné.

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
NodeType	NetType	r		Un abonné appelle son type du sous-réseau.
Address	String	r/w	Dyna- mique	

Attributs d'un abonné de type PC internal

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r		Nom de l'abonné.
NodeId	String	r		ID de l'abonné.
NodeType	NetType	r		Un abonné appelle son type du sous-réseau.

Attributs d'un abonné de type PROFIBUS

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r		Nom de l'abonné.
NodeId	String	r		ID de l'abonné.
NodeType	NetType	r		Un abonné appelle son type du sous-réseau.
Address	String	r/w	Dyna- mique	

Attributs d'un abonné de type PROFIBUS integrated

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r		Nom de l'abonné.
NodeId	String	r		ID de l'abonné.
NodeType	NetType	r		Un abonné appelle son type du sous-réseau.
Address	String	r	Dyna- mique	

Code de programme : attributs d'un abonné

Pour appeler ou définir les attributs d'un abonné, modifiez le code de programme suivant :

```
Node node = ...;
string nameValue = node.Name;
NetType nodeType = node.NodeType;
node.NodeType = NetType.Mpi;
```

Code de programme : Attributs dynamiques

Pour appeler ou définir les attributs dynamiques d'un abonné, modifiez le code de programme suivant :

```
Node node = ...;
var attributeNames = new[]
{
    "Address", "SubnetMask", "RouterAddress", "UseRouter", "DhcpClientId",
    "IpProtocolSelection"
};
foreach (var attributeName in attributeNames)
{
    object attributeValue = ((IEngineeringObject)node).GetAttribute(attributeName);
}
```

Sélection de protocole

Valeur	Description
IpProtocolSelection.None	Valeur d'erreur
IpProtocolSelection.Project	Suite IP configurée dans le projet.
IpProtocolSelection.Dhcp	Suite IP gérée à l'aide du protocole DHCP. ID du client DHCP nécessaire.
IpProtocolSelection.UserProgram	Suite IP définie à l'aide d'un FB (bloc fonctionnel).
IpProtocolSelection.OtherPath	Suite IP définie par d'autres méthodes, par exemple PST Tool.
IpProtocolSelection.VialoController	Suite IP définie à l'aide du contrôleur IO au Runtime.

Type de réseau

Valeur	Description
NetType.Asi	Il s'agit d'un réseau de type ASI.
NetType.Ethernet	Il s'agit d'un réseau de type Ethernet.
NetType.Link	Il s'agit d'un réseau de type Link.
NetType.Mpi	Il s'agit d'un réseau de type MPI.
NetType.PcInternal	Il s'agit d'un réseau de type PC Internal.
NetType.Profibus	Il s'agit d'un réseau de type PROFIBUS.
NetType.ProfibusIntegrated	Il s'agit d'un réseau du type PROFIBUS integrated.

Valeur	Description
NetType.Ptp	Il s'agit d'un réseau de type PTP.
NetType.Wan	Il s'agit d'un réseau de type Wide Area Network (WAN).
NetType.Unknown	Le type de réseau est inconnu.

7.14.11 Relier l'abonné à un sous-réseau

Condition

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour affecter un abonné (appareil, interface) à un réseau, modifiez le code de programme suivant :

```
Node node = ...;  
Subnet subnet = ...;  
node.ConnectToSubnet(subnet);
```

7.14.12 Déconnecter un abonné d'un sous-réseau

Condition

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour déconnecter un abonné (appareil, interface) d'un réseau, modifiez le code de programme suivant :

```
Node node = ...;  
node.DisconnectFromSubnet();
```

7.14.13 Créer un réseau IO

Condition

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Pour créer un réseau IO, appelez une action `IoController.CreateIoSystem("name")` sur un objet du type `IoController`. Lorsque `name` a zéro comme valeur ou `String.Empty`, c'est le nom par défaut qui est utilisé. Le contrôleur IO est détecté lors de l'accès à l'objet de l'attribut `IoControllers` sur `NetworkInterface`. Le navigateur de l'`IoController` retourne un objet `IoController`.

Conditions préalables à la création d'un réseau IO :

- L'interface du contrôleur IO est reliée à un sous-réseau.
- Le contrôleur IO ne possède aucun réseau IO.

Code du programme

Pour créer un réseau IO, modifiez le code de programme suivant :

```
using System.Linq;
...

NetworkInterface interface = ...;
IoSystem ioSystem = null;

// Interface is configured as io controller
if((interface.InterfaceOperatingMode & InterfaceOperatingModes.IoController) != 0)
{
    IoControllerComposition ioControllers = interface.IoControllers;
    IoController ioController = ioControllers.First();
    if(ioController != null)
    {
        ioSystem = ioController.CreateIoSystem("io system");
    }
}
```

7.14.14 Appeler les attributs d'un réseau IO

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Le système maître et le réseau IO sont représentés tous deux par la classe IoSystem.

Code de programme : Attributs d'un réseau IO

Pour appeler les attributs de l'IoSystem, modifiez le code de programme suivant :

```
NetworkInterface itf = ...
foreach (IIoController ioController in itf.IoControllers)
{
    IoSystem ioSystem = ioController.IoSystem;
    int ioSystemNumber = ioSystem.Number;
    string ioSystemName = ioSystem.Name;
}
```

Code de programme : sous-réseau d'un réseau IO

Pour naviguer jusqu'au sous-réseau associé au réseau IO, modifiez le code de programme suivant :

```
Subnet subnet = ioSystem.Subnet;
```

7.14.15 Relier IO-Connector à un réseau IO

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Utilisez l'action `ConnectToIoSystem(ioSystem ioSystem)` de `IoConnector` pour relier un `IoConnector` Profinet ou DP à un réseau IO existant.

Utilisez l'action `GetIoController` pour naviguer jusqu'au `IoController` décentralisé. Pour plus d'informations sur la navigation vers l'`IoConnector` local et le réseau IO, voir Appeler le système maître ou le réseau IO d'une interface (Page 191).

Conditions

- `IoConnector` n'est pas encore relié à un réseau IO.
- L'interface `IoConnector` est reliée au même sous-réseau que l'interface de l'`IoController` souhaité.

Code de programme

Modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;
IoConnector ioConnector = ...;
ioConnector.ConnectToIoSystem();
IoController ioController = ioConnector.GetIoController();
```

7.14.16 Appeler le système maître ou le réseau IO d'une interface

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Le service `NetworkInterface` fournit au navigateur des `IoControllers` et chaque `IoController` fournit à son tour un `IoSystem` au navigateur. Le système maître et le réseau IO sont représentés tous deux par la classe `IoSystem`. L'esclave et le périphérique IO sont dénommés tous deux "périphérique IO".

- Le navigateur du `IoControllers` retourne des objets d'`IoController` quand l'interface de réseau peut avoir un réseau IO. Pour l'instant, seul un contrôleur IO est retourné.
- Le navigateur du `IoConnectors` retourne des objets d'`IoConnector` quand l'interface de réseau peut être reliée à un réseau IO en tant que périphérique IO. Pour l'instant, seul un connecteur IO est retourné.

Code de programme : appeler le réseau IO du IoController

Pour appeler le réseau IO du IoController, modifiez le code de programme suivant :

```
NetworkInterface itf = ...
foreach (IoController ioController in itf.IoControllers)
{
    IoSystem ioSystem = ioController.IoSystem;
    // work with the io system
}
```

Code de programme : appeler le réseau IO du IoConnector

Pour appeler le réseau IO du IoConnector, modifiez le code de programme suivant :

```
NetInterface itf = ...
foreach (IoConnector ioConnector in itf.IoConnectors)
{
    IoSystem ioSystem = ioConnector.ConnectedIoSystem;
    // work with the io system
}
```

7.14.17 Appeler un contrôleur IO

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code de programme

Actuellement, seules des configurations avec un IoController sont possibles. Un IoController ne dispose pas d'attributs modélisés ni d'actions.

Code de programme

Pour appeler le contrôleur IO, modifiez le code de programme suivant :

```
NetworkInterface itf = ...
foreach (IoController ioController in itf.IoControllers)
{
    // work with the io controller
}
```


7.14.18 Appeler un connecteur IO

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Un IoConnector ne dispose pas d'attributs modélisés ni d'actions.

Code de programme

Pour appeler le connecteur IO, modifiez le code de programme suivant :

```
NetworkInterface itf = ...
foreach (IoConnector ioConnector in itf.IoConnectors)
{
    // work with the IoConnector
}
```

7.14.19 Déconnecter IO-Connector d'un réseau IO ou d'un réseau maître DP

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Utilisez l'action DisconnectFromIoSystem() de IoConnector pour déconnecter un IoConnector d'un réseau IO existant ou d'un réseau maître DP existant.

Pour plus d'informations sur la navigation vers l'IoConnector local et le réseau IO, voir Appeler le système maître ou le réseau IO d'une interface (Page 191).

Code de programme

Modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;
IoConnector ioConnector = ...;

ioConnector.DisconnectFromIoSystem();
```

7.14.20 Appeler un réseau maître DP

Condition

- L'application TIA Portal Openness est connectée à TIA Portal. Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert. Voir Ouvrir un projet (Page 99)

Utilisation

Un réseau maître DP possède certains attributs pouvant être lus et/ou écrits. Les attributs ne sont disponibles que s'ils sont visibles dans l'IU. En règle générale, l'opération d'écriture n'est autorisée que sur un attribut que l'utilisateur peut également modifier dans l'UI. Cela peut varier en fonction du maître DP et des esclaves DP qui sont attribués à ce réseau maître DP.

Attributs d'un réseau maître DP

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
Name	String	r/w		
Number	Int	r/w		L'attribut Number reprend les valeurs qui ne peuvent pas être définies via l'interface utilisateur. Dans un tel cas, la compilation échoue.

Code de programme : appeler des attributs

Pour appeler les attributs, modifiez le code de programme suivant :

```
IoSystem dpMastersystem = ...;

string name = dpMastersystem.Name;
int number = dpMastersystem.Number;
```

Code de programme : définir des attributs

Pour définir les attributs, modifiez le code de programme suivant :

```
IoSystem dpMastersystem = ...;

dpMastersystem.Name = "myDpMastersystem"
dpMastersystem.Number=42;
```

7.14.21 Appeler les attributs d'un réseau PROFINET IO

Condition

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Un réseau IO possède certains attributs pouvant être lus et/ou écrits. Les attributs ne sont disponibles que s'ils sont visibles dans l'IU. En règle générale, l'opération d'écriture n'est autorisée que sur un attribut que l'utilisateur peut également modifier dans l'UI. Cela peut varier en fonction du contrôleur IO et des périphériques IO qui sont attribués à ce réseau IO.

Attributs d'un réseau PROFINET IO

Attribut	Type de données	Acces- sible en écritu- re	Accès	Description
MultipleUseloSystem	Bool	r/w	Dyna- mique	
Name	String	r/w		
Number	Int	r/w		L'attribut Number reprend les valeurs qui ne peuvent pas être définies via l'interface utilisateur. Dans un tel cas, la compilation échoue.
UseloSystemNameAsDeviceNameExtension	Bool	r/w	Dyna- mique	Lorsque MultipleUseloSystem est défini sur TRUE, UseloSystemNameAsDeviceNameExtension est défini sur FALSE et l'accès en écriture n'est plus possible.
MaxNumberIWlanLinksPerSegment	Int	r/w	Dyna- mique	

Code du programme : appeler des attributs

Pour appeler les attributs, modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;  
string name = ioSystem.Name;
```

Code du programme : définir des attributs

Pour définir les attributs, modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;  
ioSystem.Name = "IOSystem_1";
```

Code du programme : appeler les attributs pour l'accès dynamique

Pour appeler les valeurs d'attributs dynamiques, modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;  
var attributeNames = new[]  
{  
    "MultipleUseIoSystem", "UseIoSystemNameAsDeviceNameExtension",  
    "MaxNumberIWlanLinksPerSegment"  
};  
foreach (var attributeName in attributeNames)  
{  
    object attributeValue = ((IEngineeringObject)ioSystem).GetAttribute(attributeName);  
}
```

Code du programme : définir les attributs pour l'accès dynamique

Pour définir les valeurs des attributs dynamiques, modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;  
((IEngineeringObject)ioSystem).SetAttribute("MultipleUseIoSystem", true);
```

7.14.22 Supprimer un réseau maître DP

Condition

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme : supprimer le réseau PROFINET IO

Pour supprimer un réseau PROFINET IO, modifiez le code de programme suivant :

```
IoController ioController = ...;  
IoSystem ioSystem = ioController.IoSystem;  
  
ioSystem.Delete();
```

7.14.23 Supprimer un réseau Profinet IO

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code de programme

Pour supprimer un réseau Profinet IO, modifiez le code de programme suivant :

```
IoController ioController = ...;  
IoSystem ioSystem = ioController.IoSystem;  
ioSystem.Delete();
```

7.14.24 Créer un réseau maître DP

Condition

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Pour créer un réseau maître DP, appelez une action `CreateIoSystem(string nameOfIoSystem)` sur un objet du type `IoController`. Le contrôleur IO est détecté lors de l'accès à l'objet de l'attribut `IoControllers` sur la `NetworkInterface`.

Conditions préalables à la création d'un réseau maître DP :

- L'interface du contrôleur IO est reliée à un sous-réseau.
- Le contrôleur IO ne possède aucun réseau IO.

Code du programme

Pour créer un réseau maître DP, modifiez le code de programme suivant :

```
using System.Linq;
...
NetworkInterface interface = ...;
IoSystem dpMasterSystem = null;

// Interface is configured as master or as master and slave
if((interface.InterfaceOperatingMode & InterfaceOperatingModes.IoController) != 0)
{
    IoControllerComposition ioControllers = interface.IoControllers;
    IoController ioController = ioControllers.First();
    if(ioController != null)
    {
        dpMasterSystem = ioController.CreateIoSystem("dp master system");
    }
}
```

7.14.25 Appeler les informations de lien des ports de l'élément d'appareil du port

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

NetworkPort offre la liaison ConnectedPorts, qui est une énumération de ports, permettant d'accéder à tous les ports partenaires connectés à un port.

Vous ne pouvez connecter ensemble que des ports pouvant également être connectés dans TIA UI. Par exemple, vous ne pouvez pas connecter ensemble deux ports d'une même interface Ethernet. Une exception récupérable est déclenchée dans les situations suivantes :

- une connexion avec le même port partenaire existe déjà,
- vous essayez de connecter deux ports qui ne peuvent pas être connectés ensemble,
- vous essayez de créer une deuxième connexion vers un port qui ne prend pas en charge d'autres partenaires.

Code de programme : appeler une connexion de port

Pour appeler les informations de lien des ports d'un l'élément d'appareil du port, modifiez le code de programme suivant :

```
NetworkPort port = ...;
foreach (NetworkPort partnerPort in port.ConnectedPorts)
{
    // work with the partner port
}
```

Code de programme : créer des connexions de port

Modifiez le code de programme suivant :

```
NetworkPort port1 = ...;
NetworkPort port2 = ...;
port1.ConnectToPort(port2);

// port supports alternative partners
NetworkPort port1 = ...;
NetworkPort port2 = ...;
NetworkPort port3 = ...;
port1.ConnectToPort(port2);
port1.ConnectToPort(port3);
```

Code de programme : supprimer une connexion de port

Modifiez le code de programme suivant :

```
NetworkPort port1 = ...;
NetworkPort port2 = ...;
port1.DisconnectFromPort(port2);
```

7.14.26 Attributs de la connexion de port

Attributs pour la connexion de port

Openness prend en charge les attributs suivants pour les connexions de port. Si les attributs sont disponibles dans l'interface utilisateur, les attributs correspondants sont également accessibles via Openness. Si l'utilisateur dispose d'un accès et peut donc modifier les attributs dans l'interface utilisateur, il peut également modifier ces derniers dans Openness.

Nom d'attribut	Type de données	Accessible en écriture	Accès	Description
MediumAttachmentType	MediumAttachmentType	Protégé en écriture	Attribut dynamique	
CableName	CableName	Lecture/écriture	Attribut dynamique	
AlternativePartner-Ports	Bool	Lecture/écriture	Attribut dynamique	Disponible uniquement si la fonction de changeur d'outil est prise en charge.
SignalDelaySelection	SignalDelaySelection	Lecture/écriture	Attribut dynamique	
CableLength	CableLength	Lecture/écriture	Attribut dynamique	
SignalDelayTime	Double	Lecture/écriture	Attribut dynamique	

Valeurs d'énumération d'attributs de connexion de port

L'énumération MediumAttachmentType a les valeurs suivantes.

Valeur	Description
MediumAttachmentType.None	Type de couplage impossible à déterminer.
MediumAttachmentType.Copper	Couplage cuivre.
MediumAttachmentType.FiberOptic	Couplage fibre optique.

L'énumération CableName a la valeur suivante.

Valeur	Description
CableName.None	Aucun nom de câble indiqué
CableName.FO_Standard_Cable_9	Câble standard FO GP (9 µm)
CableName.Flexible_FO_Cable_9	Câble FO souple (9 µm)
CableName.FO_Standard_Cable_GP_50	Câble standard FO GP (50 µm)
CableName.FO_Trailing_Cable_GP	Câble traînant FO / GP
CableName.FO_Ground_Cable	Câble de mise à la terre FO
CableName.FO_Standard_Cable_62_5	Câble standard FO (62,5 µm)
CableName.Flexible_FO_Cable_62_5	Câble FO souple (62,5 µm)
CableName.POF_Standard_Cable_GP	Câble standard POF GP
CableName.POF_Trailing_Cable	Câble traînant POF
CableName.PCF_Standard_Cable_GP	Câble traînant PCF / GP
CableName.GI_POF_Standard_Cable	Câble standard POF GI
CableName.GI_POF_Trailing_Cable	Câble traînant POF GI
CableName.GI_PCF_Standard_Cable	Câble standard PCF GI
CableName.GI_PCF_Trailing_Cable	Câble traînant PCF GI
CableName.GI_POF_Standard_Cable	Câble standard POF GI

Valeur	Description
CableName.GI_POF_Trailing_Cable	Câble traînant POF GI
CableName.GI_PCF_Standard_Cable	Câble standard PCF GI
CableName.GI_PCF_Trailing_Cable	Câble traînant PCF GI

L'énumération SignalDelaySelection a les valeurs suivantes.

Valeur	Description
SignalDelaySelection.None	
SignalDelaySelection.CableLength	CableLength sert à déterminer le retard de signal.
SignalDelaySelection.SignalDelayTime	CableDelayTime sert à déterminer le retard de signal.

L'énumération CableLength a les valeurs suivantes.

Valeur	Description
CableLength.None	La longueur de câble n'est pas indiquée.
CableLength.Length20m	Longueur de câble 20 m.
CableLength.Length50m	Longueur de câble 50 m.
CableLength.Length100m	Longueur de câble 100 m.
CableLength.Length1000m	Longueur de câble 1 000 m.
CableLength.Length3000m	Longueur de câble 3 000 m.

Attributs d'options de port

Les attributs d'options de port sont représentés ci-après.

Nom d'attribut	Type de données	Accessible en écriture	Accès
PortActivation	Bool	Lecture/écriture	Attribut dynamique
TransmissionRateAnd-Duplex	TransmissionRateAnd-Duplex	Lecture/écriture	Attribut dynamique
PortMonitoring	Bool	Lecture/écriture	Attribut dynamique
TransmissionRateAuto-Negotiation	Bool	Lecture/écriture	Attribut dynamique
EndOfDetectionOfAccessibleDevices	Bool	Lecture/écriture	Attribut dynamique
EndOfTopologyDiscovery	Bool	Lecture/écriture	Attribut dynamique
EndOfSyncDomain	Bool	Lecture/écriture	Attribut dynamique

L'énumération TransmissionRateAndDuplex a les valeurs suivantes.

Valeur	Description
TransmissionRateAndDuplex.None	
TransmissionRateAndDuplex.Automatic	Automatique
TransmissionRateAndDuplex.AUI10Mbps	AUI 10 Mbps
TransmissionRateAndDuplex.TP10MbpsHalfDuplex	TP 10 Mbps semi-duplex
TransmissionRateAndDuplex.TP10MbpsFullDuplex	TP 10 Mbps duplex intégral
TransmissionRateAndDuplex.AsyncFiber10MbpsHalfDuplex	Fibre optique asynchrone 10 Mbps semi-duplex
TransmissionRateAndDuplex.AsyncFiber10MbpsFullDuplex	Fibre optique asynchrone 10 Mbps duplex intégral
TransmissionRateAndDuplex.TP100MbpsHalfDuplex	TP 100 Mbps semi-duplex
TransmissionRateAndDuplex.TP100MbpsFullDuplex	TP 100 Mbps duplex intégral
TransmissionRateAndDuplex.FO100MbpsFullDuplex	FO 100 Mbps duplex intégral
TransmissionRateAndDuplex.X1000MbpsFullDuplex	X1000 Mbps duplex intégral
TransmissionRateAndDuplex.FO1000MbpsFullDuplexLD	FO 1 000 Mbps duplex intégral LD
TransmissionRateAndDuplex.FO1000MbpsFullDuplex	FO 1 000 Mbps duplex intégral
TransmissionRateAndDuplex.TP1000MbpsFullDuplex	TP 1 000 Mbps duplex intégral
TransmissionRateAndDuplex.FO10000MbpsFullDuplex	FO 10 000 Mbps duplex intégral
TransmissionRateAndDuplex.FO100MbpsFullDuplexLD	FO 100 Mbps duplex intégral LD
TransmissionRateAndDuplex.POFPCF100MbpsFullDuplexLD	POF/PCF 100 Mbps duplex intégral

7.14.27 Appeler les attributs d'un port

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Un élément d'appareil qui est en même temps un port offre des fonctions supplémentaires par rapport à un élément d'appareil simple.

- Il est possible d'accéder aux ports partenaires qui lui sont associés.
- Il est possible d'accéder à l'interface du port.

Pour accéder à ces fonctions supplémentaires, il faut utiliser la fonction `NetworkPort`, un service particulier de l'élément d'appareil.

Code de programme : accéder à un port

Pour accéder aux attributs d'une voie, modifiez le code de programme suivant :

```
NetworkPort port = ((IEngineeringServiceProvider) deviceItem).GetService<NetworkPort>();
if (port != null)
{
    ... // Work with the port
}
```

Attributs d'un port

Un port possède les attributs suivants :

```
NetworkPort port = ...;
var connectedPorts = port.ConnectedPorts;
var myInterface = port.Interface;
```

7.14.28 Énumérer les systèmes maître DP d'un sous-réseau

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'énumération d'un `IoSystem` fournit tous les systèmes maître DP qui se trouvent dans le sous-réseau. Le système maître et le réseau IO sont représentés tous deux par la classe `IoSystem`.

Code de programme

Pour énumérer les réseaux maître DP du sous-réseau, modifiez le code de programme suivant :

```
Subnet subnet = ...;
foreach (IoSystem ioSystem in subnet.IoSystems)
{
    // work with the io system
}
```

7.14.29 Énumérer les connecteurs IO affectés

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Le système maître et le réseau IO sont représentés tous deux par la classe IoSystem.

Est utilisé pour :

- Énumération des connecteurs IO affectés du système maître DP
- Énumération des connecteurs IO affectés du réseau Profinet IO

Code de programme

Pour énumérer les connecteurs IO affectés du réseau maître DP, modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;
foreach (IoConnector ioConnector in ioSystem.ConnectedIoDevices)
{
    // work with the io connector
}
```

7.14.30 Relier IO-Connector DP à un réseau maître DP

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Utilisez l'action `ConnectToIoSystem(ioSystem ioSystem)` de `IoConnector` pour relier un `IoConnector` à un réseau maître DP existant.

Utilisez l'action `GetIoController` pour naviguer jusqu'au `IoController` décentralisé. Pour plus d'informations sur la navigation vers l'`IoConnector` local et le réseau IO, voir Appeler le système maître ou le réseau IO d'une interface (Page 191).

Conditions

- `IoConnector` n'est pas encore relié à un réseau IO.
- L'interface `IoConnector` est reliée au même sous-réseau que l'interface de l'`IoController` souhaité.

Code de programme

Modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;
IoConnector ioConnector = ...;
ioConnector.ConnectToIoSystem(ioSystem);
IoController ioController = ioConnector.GetIoController();
```

7.15 Fonctions sur les appareils

7.15.1 Attributs obligatoires d'appareils

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Chaque appareil ou élément d'appareil possède certains attributs obligatoires pouvant être lus et/ou écrits. Ces attributs sont toujours les mêmes que dans l'interface utilisateur de TIA Portal.

Openness prend en charge les attributs suivants :

Nom d'attribut	Type de données	Accessible en écriture	Accès	Commentaire
Author	String	read/write	Dynamique	
Comment	String	read/write	Dynamique	Parfois accès protégé en écriture
IsGsd	Bool	read		VRAI quand la description d'appareil est installée avec GSD/GSDML
Name	Chaîne de caractères	read/write		Parfois accès protégé en écriture
TypeIdentfier	String	read		
TypeName	String	read	Dynamique	

Code de programme : Attributs obligatoires d'un appareil

Pour appeler les attributs obligatoires d'un appareil, modifiez le code de programme suivant :

```
Device device = ...;
string nameValue = device.Name;
bool isGsdValue = device.IsGsd;
```

Code de programme : attributs obligatoires pour accès dynamique

Pour appeler les attributs pour un accès dynamique, modifiez le code de programme suivant :

```
Device device = ...;
var attributeNames = new[] {
    "TypeName", "Author", "Comment"
};
foreach (var attributeName in attributeNames) {
    object attributeValue = ((IEngineeringObject)device).GetAttribute(attributeName);
}
```

7.15.2 Appeler l'identifiant de type des appareils et des éléments d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'attribut `TypeIdentifier` est utilisé pour identifier un objet matériel qui peut être créé avec la TIA Portal Openness API. `TypeIdentifier` est une chaîne de caractères composée de plusieurs parties : `<TypeIdentifierType>:<Identifier>`

Les valeurs possibles pour `TypeIdentifierType` sont :

- `OrderNumber`
- `GSD`
- `System`

OrderNumber

`OrderNumber` est le `TypeIdentifier` général pour tous les modules présents dans le catalogue du matériel.

Format de l'identifiant de type	Exemple	Particularités
<code><OrderNumber></code>	OrderNumber:3RK1 200-0CE00-0AA2	
<code><OrderNumber>/<FirmwareVersion></code>	OrderNumber:6ES7 510-1DJ01-0AB0/V2.0	La version du firmware est facultative quand elle n'est pas dans le système ou qu'il n'y en a qu'une seule. Veuillez noter
<code><OrderNumber>//<AdditionalTypeIdentifier></code>	OrderNumber:6AV2124-2DC01-0AX0//Landscape	L'identifiant de type supplémentaire peut être nécessaire quand <code>OrderNumber</code> et <code>FirmwareVersion</code> ne donnent pas de résultat univoque dans le système.

Remarque

Pour un petit nombre de modules dans le catalogue du matériel, le numéro d'article contient des caractères de remplacement qui représentent un certain groupe de matériel réel, comme par ex. les différentes longueurs des châssis de S7-300. Dans ce cas, vous pouvez utiliser non seulement l'`OrderNumber`, mais aussi l'`OrderNumber` avec caractères de remplacement pour créer une instance de l'objet matériel. Mais les caractères de remplacement ne doivent pas figurer à n'importe quelle position.

GSD

C'est l'identifiant utilisé pour les modules qui sont ajoutés à TIA Portal au moyen de GSD ou de GSDML.

Format de l'identifiant de type	Exemple	Particularités
<code><GsdName>/<GsdType></code>	GSD:SIEM8139.GSD/DAP	<p><code>GsdName</code> est le nom de GSD ou de GSDML en majuscules.</p> <p><code>GsdType</code> est l'un des suivants :</p> <ul style="list-style-type: none"> • D : appareil (device) • R : châssis (rack) • DAP : module de tête • M : module • SM : sous-module <p><code>GsdId</code> est l'identifiant de type.</p>
<code><GsdName>/<GsdType>/<GsdId></code>	GSD:SIEM8139.GSD/M/4	

System

C'est l'identifiant pour les objets ne pouvant être déterminés au moyen de `OrderNumber` ou de `GSD`.

Format de l'identifiant de type	Exemple	Particularités
<code><SystemTypeIdentifier></code>	System:Device.S7300	<code>SystemTypeIdentifier</code> est l'identifiant primaire d'un objet.
<code><SystemTypeIdentifier>/ <AdditionalTypeIdentifier></code>	GSD:SIEM8139.GSD/ M/4	<code>AdditionalTypeIdentifier</code> peut être nécessaire quand <code>SystemTypeIdentifier</code> n'est pas unique. Les préfixes pour certains types d'objet sont : <ul style="list-style-type: none"> • Connection. • Subnet. • Device. • Rack.

Code de programme

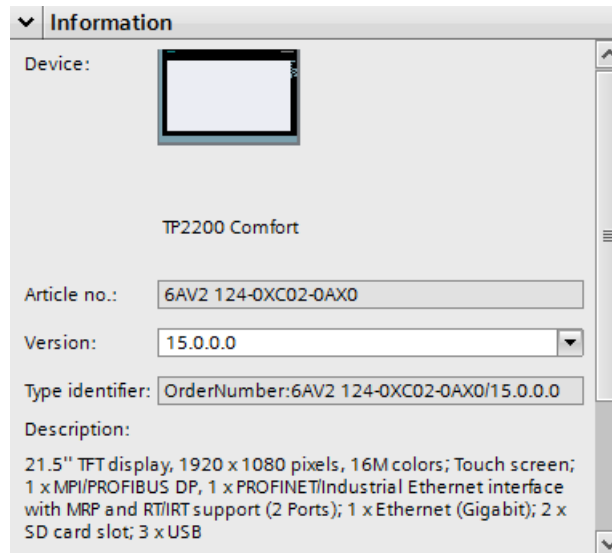
Pour appeler l'identifiant de type des objets pour GSD que l'utilisateur peut créer séparément et gérer, modifiez le code de programme suivant :

```
HardwareObject hardwareObject = ...;
string typeIdentifier = hardwareObject.TypeIdentifier;
```

Afficher les identifiants de type dans TIA Portal

Lorsque vous devez connaître un identifiant de type, vous le déterminez comme suit dans TIA Portal :

1. Activez le paramètre "Affichage de l'identifiant de type pour les appareils et les modules" sous "Options > Paramètres > Configuration matérielle > Affichage de l'identifiant de type".
2. Ouvrez l'éditeur "Appareils & Réseaux".
3. Sélectionnez un appareil dans le catalogue.
L'identifiant de type s'affiche sous "Information".



7.15.3 Créer un appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Il y a deux manières de créer un appareil, dans un projet ou dans un groupe d'appareils :

- Créer un appareil au moyen d'un identifiant de type d'élément d'appareil comme dans le catalogue du matériel de TIA

```
Device CreateWithItem(DeviceItemId, DeviceItemName, DeviceName)
```
- Créer uniquement l'appareil

```
Device Create(DeviceTypeId, DeviceName)
```

Nom	Type	Description
DeviceItemTypeId	String	Identifiant de type de l'élément d'appareil
DeviceTypeId	String	Identifiant de type de l'appareil
DeviceItemName	String	Nom de l'élément d'appareil créé
DeviceName	String	Nom de l'appareil créé

Voir : Identifiant de type (Page 207)

Code de programme : créer un appareil avec l'identifiant de type

Pour créer un objet appareil au moyen d'un identifiant de type, modifiez le code suivant :

```
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");
```

Code de programme : créer uniquement un appareil

Pour ne créer que l'objet appareil, modifiez le code suivant :

```
DeviceComposition devices = ...;
Device deviceOnly = devices.Create("System:Device.S7300", "S7300Device");
Device gsdDeviceOnly = devices.Create("GSD:SIEM8139.GSD/D", "GSD Device");
```

7.15.4 Énumérer des appareils

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation : énumérer des appareils

La TIA Portal Openness API classe les appareils à l'instar de la navigation du projet dans TIA Portal. PNV

- Les appareils qui sont directement subordonnés à un projet sont classés par groupes dans la composition "Devices" du projet.
- Les appareils qui se trouvent dans des dossiers d'appareils sont classés par groupes dans la composition "Devices" du dossier.

Remarque

Tenir compte de Hiérarchie des objets matériels du modèle d'objet (Page 64).

Utilisez l'une des possibilités suivantes pour énumérer les appareils d'un projet :

- Énumérer tous les appareils du premier niveau
- Énumérer tous les appareils en groupes ou sous-groupes
- Énumérer tous les appareils d'un projet ne contenant aucun groupe d'appareils
- Énumérer tous les appareils des groupes système d'appareils non groupés

Exemples d'appareils pouvant être énumérés :

- Central station
- PB-Slave / PN-IO device
- HMI Device

Code de programme : énumérer les appareils du premier niveau

Pour énumérer les appareils du premier niveau, utilisez le code de programme suivant :

```
private static void EnumerateDevicesInProject(Project project)
{
    DeviceComposition deviceComposition = project.Devices;
    foreach (Device device in deviceComposition)
    {
        // add code here
    }
}
```

Pour accéder à un appareil en particulier, modifiez le code de programme suivant :

```
private static void AccessSingleDeviceByName(Project project)
{
    DeviceComposition deviceComposition = project.Devices;
    // The parameter specifies the name of the device
    Device device = deviceComposition.Find("MyDevice");
}
```

Code de programme : énumérer des appareils en groupes ou sous-groupes

Pour accéder aux appareils d'un groupe, vous devez d'abord naviguer jusqu'au groupe, puis jusqu'à l'appareil.

Modifiez le code de programme suivant :

```
//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
    }
}
```

Code de programme : trouver des appareils spécifiques

Pour rechercher un appareil en particulier avec son nom, modifiez le code de programme suivant :

```
//Find a specific device by name
Project project = ...
Device plc1 = project.Devices.First(d => d.Name == "Mydevice");
... // Work with the device
```

Pour rechercher un appareil en particulier avec la méthode "Find", modifiez le code de programme suivant :

```
//Find a specific device via "Find" method
Project project = ...
Device plc1 = project.Devices.Find("MyDevice");
... // Work with the device
```

Code de programme : énumérer les appareils d'un projet ne contenant aucun groupe d'appareils

Modifiez le code de programme suivant :

```
//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
}
```

Code de programme : énumérer tous les appareils dans un dossier

Modifiez le code de programme suivant :

```
//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
```

Code de programme : énumérer les appareils des groupes système d'appareils non groupés

Pour structurer les projets, des appareils décentralisés ont été ajoutés dans le groupe UngroupedDevices. Pour accéder à ce groupe, naviguez d'abord jusqu'au groupe, puis jusqu'à l'appareil.

Modifiez le code de programme suivant :

```
//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
```

7.15.5 Accéder à des appareils

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Chaque appareil basé sur GSD ou sur GSDML dispose d'attributs. Certains servent à la détermination du type exact de l'appareil.

Nom	Type de données	Accessible en écriture	Accès	Description
Author	String	read/write	Dynamique	
Comment	String	read/write	Dynamique	
GsdName	String	read	Dynamique	Nom du fichier GSD ou GSDML.
GsdType	String	read	Dynamique	Type de l'objet matériel. La valeur de l'appareil est toujours "D".
GsdId	String	read	Dynamique	Identifiant spécifique de l'objet matériel. Toujours vide pour les appareils.
IsGsd	Bool	read		VRAI pour appareil GSD ou appareil GSDML
Name	String	read/write		
TypeIdentifiant	String	read		

Code de programme : appeler les attributs d'identification

Pour appeler les attributs, modifiez le code de programme suivant :

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId"
};
foreach (var attributeName in attributeNames) {
    object attributeValue = device.GetAttribute(attributeName);
}
```

Code de programme : Attributs

Pour appeler les attributs, modifiez le code de programme suivant :

```
Device device = ...;
string nameValue = device.Name;
bool isGsdValue = device.IsGsd;
```

Code de programme : attributs pour accès dynamique

Pour appeler les attributs, modifiez le code de programme suivant :

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId"
};
foreach (var attributeName in attributeNames) {
    object attributeValue = device.GetAttribute(attributeName);
}
```

Particularités des appareils GSD

Un appareil qui est en même temps appareil GSD offre des fonctions supplémentaires. Pour appeler la fonction GsdDevice, utilisez la méthode GetService.

```
GsdDevice gsdDevice = ((IEngineeringServiceProvider)deviceItem).GetService<GsdDevice>();
if (gsdDevice != null) {
    ... // work with the GSD device
};
```

Code de programme : attributs d'un appareil GSD

Pour appeler les attributs, modifiez le code de programme suivant :

```
Device device = ...;
GsdDevice gsdDevice = ...;
string gsdId = gsdDevice.GsdId;
string gsdName = gsdDevice.GsdName;
string gsdType = gsdDevice.GsdType;
bool isProfibus = gsdDevice.IsProfibus;
bool isProfinet = gsdDevice.IsProfinet;
```

7.15.6 Supprimer un appareil**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code de programme

Pour supprimer un appareil, modifiez le code de programme suivant :

```
Project project = ...;  
Device deviceToDelete = project.UngroupedDevices.Devices.Find(".....");  
  
// delete device  
deviceToDelete.Delete();
```

7.16 Fonctions d'éléments d'appareils

7.16.1 Attributs obligatoires des éléments d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Chaque appareil ou élément d'appareil possède certains attributs obligatoires pouvant être lus et/ou écrits. Ces attributs sont toujours les mêmes que dans l'interface utilisateur de TIA Portal.

Les attributs suivants sont pris en charge dans TIA Portal Openness :

Nom d'attribut	Type de données	Acces- sible en écriture	Accès	Commentaire
Author	String	read/write	Dynami- que	
Classification	DeviceItemClassi- fication	read		
Comment	String	read/write	Dynami- que	Parfois accès protégé en écriture
FirmwareVersion	String	read	Dynami- que	
InterfaceOpera- tingMode	InterfaceOpera- tingModes	read/write	Dynami- que	Pour éléments d'appareil avec la fonction <code>NetworkInterface</code>
InterfaceType	NetType	readwrite	Dynami- que	Pour éléments d'appareil avec la fonction <code>NetworkInterface</code>
IsBuiltIn	Bool	read		FAUX pour les objets pouvant être créés par l'utilisateur
IsGsd	Bool	read		VRAI quand la description d'appareil est installée avec GSD/GSDML
IsPlugged	Bool	read		VRAI pour les appareils enfichés
Label	String	read	Dynami- que	Pour les éléments d'appareil avec la fonction <code>NetworkPort</code> ou <code>NetworkInterface</code> . Lorsque l'interface ou le port ne possède pas d'attribut "La- bel", alors <code>Label</code> est <code>String.Empty</code> .
LocationIdentifier	String	read/write	Dynami- que	
Name	String	read/write		Parfois accès protégé en écriture
OrderNumber	String	read/write	Dynami- que	Parfois accès protégé en écriture

Nom d'attribut	Type de données	Acces- sible en écriture	Accès	Commentaire
PlantDesignation	String	read/write	Dynami- que	
PositionNumber	int	read		
TypeIdentifier	String	read		
TypeName	String	read	Dynami- que	Le nom de type indépendant de la langue. (Optionnel pour les éléments d'appareil que l'utilisateur ne peut pas gérer en tant qu'éléments automatiquement créés ou sous-modules fixes).

Classification de l'élément d'appareil

Valeur	Description
DeviceItemClassifications.None	Pas de classification.
DeviceItemClassifications.CPU	L'élément d'appareil est une CPU.
DeviceItemClassifications.HM	L'élément d'appareil est un module de tête.

Code de programme : attributs obligatoires d'un élément d'appareil

Pour appeler les attributs obligatoires d'un élément d'appareil, modifiez le code de programme suivant :

```
DeviceItem deviceItem = ...;
string nameValue = deviceItem.Name;
string typeIdentifierValue = deviceItem.TypeIdentifier;
int positionNumberValue = deviceItem.PositionNumber;
bool isBuiltInValue = deviceItem.IsBuiltIn;
bool isPluggedValue = deviceItem.IsPlugged;
```

Code de programme : attributs obligatoires pour accès dynamique

Pour appeler les attributs pour un accès dynamique, modifiez le code de programme suivant :

```
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.");
```

7.16.2 Créer et enficher un élément d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'action `PlugNew(string typeIdentiflier, string name, int positionNumber)` de `HardwareObject` est utilisée pour

- créer un nouvel élément d'appareil et l'enficher dans un objet matériel existant,
- créer un nouveau sous-élément d'appareil, par exemple un sous-module et l'enficher dans un élément d'appareil existant.

Quand l'action réussit, elle fournit l'objet "élément d'appareil" créé. Sinon, une exception récupérable est lancée.

Avec l'action `CanPlugNew(string typeIdentiflier, string name, int positionNumber)`, vous pouvez déterminer si la création et l'enfichage seront possibles. Quand son exécution n'est pas possible, l'action fournit `false`.

L'action peut échouer pour les raisons imprévisibles suivantes, même si la méthode retourne la valeur `True`.

- un numéro de position est déjà utilisé par un autre élément d'appareil,
- l'élément d'appareil actuel ne peut pas être enfiché à la position alors quelle est libre,
- le conteneur ne met pas le numéro de position à disposition,
- le nom de l'élément d'appareil est déjà utilisé par un autre élément d'appareil existant dans le même conteneur,
- l'élément d'appareil ne peut pas être enfiché dans le conteneur,
- l'appareil est en ligne.

Le tableau ci-dessous indique les paramètres de méthode requis :

Nom	Type	Description
<code>typeIdentiflier</code>	String	Identifiant de type de l'élément d'appareil créé
<code>name</code>	String	Nom de l'élément d'appareil créé
<code>positionNumber</code>	int	Numéro de position de l'élément d'appareil créé

Code de programme

Pour enficher un élément d'appareil dans un objet matériel existant, modifiez le code de programme suivant :

```
HardwareObject hwObject = ...;
string typeIdentifier = ...;
string name = ...;
int positionNumber = ...;
if(hwObject.CanPlugNew(typeIdentifier, name, positionNumber))
{
    DeviceItem newPluggedDeviceItem = hwObject.PlugNew(typeIdentifier, name,
positionNumber);
}
```

Accès aux informations sur les modules

Un utilisateur d'Openness peut accéder à des informations sur les modules enfichables via l'objet ModuleInformationProvider. L'utilisateur a accès

- aux types de conteneur dans lesquels un module donné doit être enfiché (par ex. un appareil ou un châssis) – via la méthode FindContainerTypes ;
- aux versions disponibles d'un module donné indiqué en partie – via la méthode FindModuleTypes.

Code de programme : accès à l'objet ModuleInformationProvider.

```
Project project = ...;
HardwareUtilityComposition extensions = project.HwUtilities;
var result = extensions.Find("ModuleInformationProvider") as ModuleInformationProvider;
```

Code de programme : accès à des types de conteneur avec la méthode FindContainerTypes

La méthode FindContainerTypes renvoie les types de conteneur d'un module indiqué. Le module est indiqué via le paramètre typeIdentifier. La liste qui en résulte contient des identificateurs de type (Typentifier) de tous les types de conteneur du type demandé. Elle comprend généralement un appareil et un châssis, et les conteneurs sont indiqués dans leur ordre dans la hiérarchie dans le projet en commençant à l'appareil.

Nom du paramètre	Type	Description
typeIdentifier	String	Identifiant de type d'un élément d'appareil

IMPORTANT

Cette méthode fonctionne uniquement pour les modules visibles dans la vue de réseau.

Cette méthode fonctionne uniquement pour les modules et non pour les sous-modules.

```
string typeIdentifieur = ...;
string[] containerTypes = moduleInformationProvider.FindContainerTypes(typeIdentifieur);
```

Code de programme : accès à des versions avec la méthode FindModuleTypes

La méthode FindModuleTypes renvoie toutes les versions possibles d'un objet matériel avec l'utilisation de l'identifiant de type partiel de l'élément d'appareil. Cette méthode permet de sortir une liste de chaînes de caractères. Chaque chaîne de caractères correspond au Typelidentifieur complet d'une correspondance possible pour un Typelidentifieur partiel.

Un identifiant de type partiel ne peut comprendre que des parties complètes, chaque partie de l'identifiant de type étant séparée par le caractère "/". Les caractères génériques ou les parties incomplètes ne sont pas pris en charge. Vous devez également tenir compte des limitations suivantes concernant le nombre minimum de parties indiquées :

- N° de référence : au moins une partie. Exemple : OrderNumber:6ES7 317-2EK14-0AB0
- GSD : au moins deux parties. Exemple : GSD:SI05816A.GSD/M
- Système : au moins une partie. Exemple : System:Rack.ET200SP

Nom du paramètre	Type	Description
partialTypeIdentifieur	String	Identifiant de type partiel d'un élément d'appareil

```
string partialTypeIdentifieur = ...;
string[] moduleTypes = moduleInformationProvider.FindModuleTypes(partialTypeIdentifieur);
```

Code de programme : accès aux emplacements d'enfichage avec la méthode GetPlugLocations

La méthode GetPlugLocations renvoie des informations sur les emplacements d'enfichage, notamment l'emplacement, le numéro de position (désignation d'un emplacement) et les emplacements libres pour l'objet matériel.

La classe PlugLocation présente les caractéristiques suivantes.

Nom de la propriété	Type	Description
PositionNumber	Int	Le numéro de position de l'emplacement libre
Label	String	La désignation de l'emplacement libre

- Si aucune "label" n'est disponible pour un numéro de position spécifique, la représentation en chaîne de caractères du numéro de position est utilisée.
- Les objets `PlugLocation` sont mis à disposition que pour des emplacements libres.

```

IHardwareObject hardwareObject = ...;
IList<PlugLocation> result = hardwareObject.GetPlugLocations();
foreach (PlugLocation item in result)
{
    Console.WriteLine("{0} - {1}", item.PositionNumber, item.Label);
}

```

7.16.3 Déplacer un élément d'appareil dans un autre emplacement d'enfichage

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Avec l'action `PlugMove(DeviceItem deviceItem, int positionNumber)` de `HardwareObject`, vous pouvez déplacer un élément d'appareil existant pour l'enficher dans un objet matériel existant. La méthode `PlugMove` ajoute les éléments d'appareil à l'endroit où le module n'a pas pu enficher l'interface utilisateur. Dans ce cas, l'action `PlugMove` est terminée avec des erreurs de compilation.

L'action `CanPlugMove(DeviceItem deviceItem, int positionNumber)` sert à déterminer la possibilité de déplacement. Si le déplacement n'est pas possible, `CanPlugMove` fournit la valeur `False` en retour. L'action peut échouer pour les raisons imprévisibles suivantes, même si la méthode retourne la valeur `True`.

- un numéro de position est déjà utilisé par un autre élément d'appareil,
- l'élément d'appareil actuel ne peut pas être enfiché dans l'emplacement alors qu'il est libre,
- le conteneur ne met pas le numéro de position à disposition,
- le nom de l'élément d'appareil est déjà utilisé par un autre élément d'appareil existant dans le même conteneur,

7.16 Fonctions d'éléments d'appareils

- l'élément d'appareil ne peut pas être enfiché dans le conteneur,
- l'élément d'appareil ne peut pas être enfiché par l'utilisateur,
- l'élément d'appareil ne peut pas être supprimé par l'utilisateur,
- l'appareil est en ligne.

Code de programme

Modifiez le code de programme suivant :

```
HardwareObject hwObject = ...;
DeviceItem deviceItemToMove = ...;
int positionNumber = ...;
if(hwObject.CanPlugMove(deviceItemToMove, positionNumber)
{
    DeviceItem movedDeviceItem = hwObject.PlugMove(deviceItemToMove, positionNumber);
}
```

7.16.4 Copier l'élément d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Utilisez l'action PlugCopy(DeviceItem deviceItem, int positionNumber) de HardwareObject pour copier un appareil au sein d'un projet et l'enficher dans du matériel existant. Dans de rares cas, la méthode PlugCopy peut fonctionner même si un module ne peut pas être enfiché dans UI. Dans ce cas, des erreurs de compilation surviennent après l'opération de copie. Quand PlugCopy réussit, la copie de l'objet élément d'appareil est fournie en retour. Sinon, une exception récupérable est lancée.

Causes possibles de l'échec d'une action :

- un numéro de position est déjà utilisé par un autre élément d'appareil,
- l'élément d'appareil actuel ne peut pas être enfiché dans l'emplacement alors qu'il est libre,
- le conteneur ne met pas le numéro de position à disposition,
- le nom de l'élément d'appareil est déjà utilisé par un autre élément d'appareil existant dans le même conteneur,
- l'élément d'appareil ne peut pas être enfiché dans le conteneur,

- l'élément d'appareil ne peut pas être enfiché dans UI,
- ...

Avec l'action `CanPlugCopy(DeviceItem deviceItem, int positionNumber)`, vous pouvez déterminer si l'opération de copie sera possible. Quand l'opération de copie ne peut pas être exécutée, `CanPlugCopy` fournit la valeur `False` en retour. Toutefois, l'action peut échouer pour des raisons imprévisibles même si la méthode retourne la valeur `True`.

Nom du paramètre	Type	Description
<code>deviceItem</code>	<code>DeviceItem</code>	Élément d'appareil à copier
<code>positionNumber</code>	<code>Int</code>	Numéro de position pour la copie de l'élément d'appareil

Code de programme

Modifiez le code de programme suivant :

```
HardwareObject hwObject = ...;
DeviceItem deviceItemToCopy = ...;
int positionNumber = ...;
if(hwObject.CanPlugCopy(deviceItemToCopy, positionNumber))
{
    DeviceItem copiedDeviceItem = hwObject.PlugCopy(deviceItemToCopy, positionNumber);
}
```

7.16.5 Supprimer un élément d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code de programme

Pour supprimer un élément d'appareil, modifiez le code de programme suivant :

```
Project project = ...;
var device = project.UngroupedDevicesGroup.Devices.Find(".....");
var deviceItem = deviceItem.DeviceItems.First();

// delete device item
deviceItem.Delete();
```

7.16.6 Énumérer des éléments d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Pour appeler un élément d'appareil, utilisez le `HardwareObject`. Les éléments d'un objet matériel sont les parties enfichées dans l'objet matériel que l'utilisateur peut voir dans TIA Portal :

- châssis dans un appareil
- module dans un châssis
- sous-module dans un module
- sous-module dans un sous-module

Remarque

Pour plus d'informations à ce sujet, voir la rubrique Hiérarchie des objets matériels du modèle d'objet (Page 64).

Code de programme : énumérer les éléments d'un appareil

Pour énumérer les éléments d'appareil d'un objet matériel, modifiez le code de programme suivant :

```
private static void EnumerateDeviceItems(HardwareObject hardwareObject)
{
    foreach (DeviceItem deviceItem in hardwareObject.Items)
    {
        // add code here
    }
}
```

Code de programme : énumérer au moyen de la hiérarchie de composition

Pour énumérer les éléments d'un appareil au moyen de la hiérarchie de composition, modifiez le code de programme suivant :

```
//Enumerates devices using an composition
private static void EnumerateDeviceItems(Device device)
{
    DeviceItemComposition deviceItemComposition = device.DeviceItems;
    foreach (DeviceItem deviceItem in deviceItemComposition)
    {
        // add code here
    }
}
```

Code de programme : énumérer des éléments d'appareils avec affectation

Pour énumérer des éléments d'appareils au moyen d'une affectation, modifiez le code de programme suivant :

```
//Enumerates devices using an association
private static void EnumerateDeviceItemsWithAssociation(Device device)
{
    DeviceItemAssociation deviceItemAssociation = device.Items;
    foreach (DeviceItem deviceItem in deviceItemAssociation)
    {
        // add code here
    }
}
```

7.16.7 Appeler des éléments d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation : appeler des éléments d'appareil

Pour appeler des objets de type "DeviceItem", utilisez les attributs suivants :

- Nom (string) : nom de l'élément d'appareil
- Conteneur (HardwareObject) : conteneur dans lequel l'élément d'appareil est enfiché

Nom	Type de données	Accessible en écriture	Accès	Description
Author	String	read/write	Dynamique	
Comment	String	read/write	Dynamique	
FirmwareVersion	String	read	Dynamique	Uniquement pour modules de tête
GsdName	String	read	Dynamique	Nom du fichier GSD.
GsdType	String	read	Dynamique	Type de l'objet matériel. La valeur de l'appareil est toujours "D".
GsdId	String	read	Dynamique	Identifiant spécifique de l'objet matériel. Toujours vide pour les appareils.
IsBuiltIn	Bool	read		
IsGsd	Bool	read		VRAI pour appareil GSD ou appareil GSDML
IsPlugged	Bool	read		
IsProfibus	Bool	read		
IsProfinet	Bool	read		
Name	String	read/write		
OrderNumber	String	read	Dynamique	Uniquement pour modules de tête
PositionNumber	Bool	read		
TypeIdentifier	String	read		

Code de programme : appeler un élément d'appareil

Pour appeler un élément d'appareil, modifiez le code de programme suivant :

```
public static DeviceItem AccessDeviceItemFromDevice(Device device)
{
    DeviceItem deviceItem = device.DeviceItems[0];
    return deviceItem;
}
```

Code de programme : accéder à un élément d'appareil d'un élément d'appareil

Pour accéder à un élément d'appareil d'un élément d'appareil, modifiez le code de programme suivant :

```
public static DeviceItem AccessDeviceItemFromDeviceItem(DeviceItem deviceItem)
{
    DeviceItem subDeviceItem = deviceItem.DeviceItems[0];
    return subDeviceItem;
}
```

Code de programme : naviguer jusqu'au conteneur d'un élément d'appareil

Modifiez le code de programme suivant avec l'attribut "Container" de DeviceItem pour retourner au conteneur d'un élément d'appareil :

```
DeviceItem deviceItem = ...;
HardwareObject container = deviceItem.Container;
```

Code de programme : appeler les attributs d'identification

Pour appeler les attributs, modifiez le code de programme suivant :

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId" };
foreach (var attributeName in attributeNames) {
    object attributeValue = ((IEngineeringObject)deviceItem).GetAttribute(attributeName);
}
```

Code de programme : appeler des attributs

Pour appeler les attributs, modifiez le code de programme suivant :

```
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;;
```

Code de programme : appeler les attributs pour l'accès dynamique

Pour appeler les attributs, modifiez le code de programme suivant :

```
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);
}
```

Code de programme : définir des attributs

Pour définir les attributs, modifiez le code de programme suivant :

```
DeviceItem deviceItem = ...;
((IEngineeringObject)deviceItem).SetAttribute("Comment", "This is a comment.");
```

Code de programme : appeler les données prm d'un module de tête

Pour appeler les données prm, modifiez le code de programme suivant :

```
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);
```

Code de programme : paramétrer les données prm d'un module de tête

Pour paramétrer les données prm, modifiez le code de programme suivant :

```
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);
```

Voir aussi

Hiérarchie des objets matériels du modèle d'objet (Page 64)

7.16.8 Accès à l'élément d'appareil en tant qu'interface**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Si un élément d'appareil est une interface, il offre des fonctions supplémentaires par rapport à un élément d'appareil simple. Cette interface permet à l'utilisateur d'accéder aux abonnés et au mode de fonctionnement de l'interface. Grâce à cette fonctionnalité, l'élément d'appareil peut être utilisé comme `IoDevice` (esclave) ou comme `IoController` (maître) via la fonction `NetworkInterface` (un service spécifique de l'élément d'appareil).

L'énumération `InterfaceOperatingModes` permet d'accéder aux propriétés de l'interface.

Valeur	Description
<code>InterfaceOperatingModes.None</code>	Valeur par défaut
<code>InterfaceOperatingModes.IoDevice</code>	Mode de fonctionnement de l'interface "IoDevice" (esclave).
<code>InterfaceOperatingModes.IoController</code>	Mode de fonctionnement de l'interface "IoController" (maître).
<code>InterfaceOperatingModes.IoDevice</code> or <code>InterfaceOperatingModes.IoController</code>	Mode d'interface : les deux modes ci-dessus.

Code de programme : accès à la fonction d'interface réseau

Pour appeler la fonction d'interface réseau, modifiez le code de programme suivant :

```
NetworkInterface itf =
((IEngineeringServiceProvider)deviceItem).GetService<NetworkInterface>();
if (itf != null)
{
... // work with the interface
}

//Accessing nodes and operating mode
NodeComposition nodes = itf.Nodes;
InterfaceOperationModes mode = itf.InterfaceOperatingMode;

//Accessing the type of interface
NetType itfType = itf.InterfaceType;

//Modifying the operating mode and interface type
itf.InterfaceOperatingMode = InterfaceOperationModes.IoDevice;
itf.InterfaceType = NetType.Profibus

//Accessing the ports linked to an interface.
NetworkPortAssociation nodes = itf.Ports;
```

7.16.9 Accéder aux attributs d'une interface d'appareil I/O

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne pour l'accès en écriture.

Utilisation

L'interface TIA Portal Openness API vous permet d'appeler ou de déterminer des attributs pour IRT et le mode isochrone sur l'interface d'appareil IO.

Accès à l'interface d'un contrôleur IO

Les attributs suivants permettent d'accéder à l'interface d'un contrôleur IO. Le contrôleur doit être le maître de synchronisation :

Nom d'attribut	Type de données	Accessible en écriture	Accès	Description
PnSendClock	Int64	r/w	Attribut dynamique	Envoyer la cadence en nano-secondes

Accès à l'interface d'un réseau IO

Les attributs suivants permettent d'accéder à l'interface d'un réseau IO. Les valeurs Ti/To peuvent être utilisées par tous les modules et sous-modules faisant partie du réseau IO.

Nom d'attribut	Type de données	Accessible en écriture	Accès
IsochronousTiToAutoCalculation	BOOL	r/w	Attribut dynamique
IsochronousTi	DOUBLE	r/w	Attribut dynamique
IsochronousTo	DOUBLE	r/w	Attribut dynamique

Accès à l'interface d'un périphérique IO

Les attributs suivants permettent d'accéder à l'interface d'un périphérique IO. Les valeurs Ti/To peuvent être utilisées par tous les modules et sous-modules faisant partie du réseau IO.

Nom d'attribut	Type de données	Accessible en écriture	Accès
IsochronousMode	BOOL	r/w	Attribut dynamique
IsochronousTiToCalculationMode	IsochronousTiToCalculationMode	r/w	Attribut dynamique
IsochronousTi	DOUBLE	r/w	Attribut dynamique
ochronousTo	DOUBLE	r/w	Attribut dynamique

Les valeurs ENUM suivantes sont disponibles pour l'attribut IsochronousTiToCalculationMode :

Valeur	Description
IsochronousTiToCalculationMode.None	
IsochronousTiToCalculationMode.FromOB	Les valeurs Ti/To de l'OB (configurées sur le réseau IO) sont utilisées.
IsochronousTiToCalculationMode.FromSubnet	Cette valeur n'est pas utilisée par les interfaces PROFINET.
IsochronousTiToCalculationModeAutomaticMinimum	Les valeurs Ti/To sont calculées automatiquement pour le périphérique IO.
IsochronousTiToCalculationMode.Manual	L'utilisateur peut saisir des valeurs Ti/To manuellement pour ce périphérique IO.

Code de programme : appeler ou déterminer les attributs d'une interface d'appareil IO

Pour accéder à la cadence d'émission, modifiez le code de programme suivant :

```
DeviceItem pnInterface = ...;

// read attribute
long attributeValue = (long)pnInterface.GetAttribute("PnSendClock");
// write attribute
long sendClock = 2000000;
pnInterface.SetAttribute("PnSendClock", sendClock);
```

Pour accéder aux valeurs Ti/To d'un OB, modifiez le code de programme suivant :

```
IoSystem ioSystem = ...;
bool titoAutoCalculation = (bool)ioSystem.GetAttribute("IsochronousTiToAutoCalculation");
ioSystem.SetAttribute("IsochronousTiToAutoCalculation", true);
```

Pour accéder au réglage isochrone d'une interface d'appareil IO, modifiez le code de programme suivant :

```
DeviceItem pnInterface = ...;
bool isochronousMode = (bool)pnInterface.GetAttribute("IsochronousMode");
pnInterface.SetAttribute("IsochronousMode", true);
```

7.16.10 Accès aux attributs de l'loController**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne pour l'accès en écriture.

Utilisation

L'interface TIA Portal Openness API vous permet d'appeler ou de déterminer des attributs de l'loController. Les attributs suivants sont disponibles sous PROFINET loController (sous une interface PROFINET). Si l'utilisateur peut modifier un attribut dans l'interface utilisateur, il peut également modifier l'attribut correspondant via Openness.

Nom d'attribut	Type de données	Type	Accès	Description
SyncRole	SyncRole	Lecture/écriture	Attribut dynamique	
PnDeviceNumber	Int	Protégé en écriture	Attribut dynamique	Cette propriété est disponible dans la TIA Portal UI sous le nœud Ethernet (section PROFINET)

La propriété Synchronisation role est disponible dans l'interface PROFINET de la TIA Portal UI. L'Enum SyncRole a les valeurs suivantes :

Valeur d'énumération	Valeur numérique
SyncRole.NotSynchronized	0
SyncRole.SyncMaster	1
SyncRole.SyncSlave	2
SyncRole.RedundantSyncMaster	4

Code de programme : définir les attributs de l'ioController

```
IoController ioController= ...;
SyncRole syncRole = (SyncRole)((IEngineeringObject)ioController).GetAttribute("SyncRole");
((IEngineeringObject)ioController).SetAttribute("SyncRole", SyncRole.SyncMaster);
```

7.16.11 Accès aux attributs de l'ioConnector

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne pour l'accès en écriture.

Utilisation

L'interface TIA Portal Openness API vous permet d'appeler ou de déterminer des attributs de l'loConnector. Les attributs suivants sont disponibles sous PROFINET IoController (sous une interface PROFINET). Si l'utilisateur peut modifier un attribut dans l'interface utilisateur, il peut également modifier l'attribut correspondant via Openness.

Il existe quatre types d'attribut tels que des attributs pour le temps d'actualisation, des attributs pour le temps de surveillance, des attributs pour la synchronisation et des attributs pour le numéro d'appareil.

Attributs pour le temps d'actualisation

Les attributs pour le temps d'actualisation sont indiqués ci-dessous.

Nom d'attribut	Type de données	Type	Accès	Description
PnUpdateTimeAutoCalculation	Bool	Lecture/écriture	Attribut dynamique	Si cet attribut est vrai, le temps d'actualisation est calculé automatiquement.
PnUpdateTime	Int64	Lecture/écriture	Attribut dynamique	Le temps d'actualisation est mesuré en nanosecondes.
PnUpdateTimeAdaption	Bool	Lecture/écriture	Attribut dynamique	

Attributs pour le temps de surveillance

Les attributs pour le temps de surveillance sont indiqués ci-dessous.

Nom d'attribut	Type de données	Type	Accès	Description
PnWatchdogFactor	Int32	Lecture/écriture	Attribut dynamique	
PnWatchdogTime	Int64	Protégé en écriture	Attribut dynamique	Le temps de surveillance est mesuré en nanosecondes.

Attributs pour la synchronisation

Les attributs pour la synchronisation sont indiqués ci-dessous.

Nom d'attribut	Type de données	Type	Accès	Description
RtClass	RtClass	Lecture/écriture	Attribut dynamique	
SyncRole	SyncRole	Protégé en écriture	Attribut dynamique	

L'énumération RtClass a les valeurs suivantes.

Valeur d'énumération	Valeur numérique
RtClass.None	0
RtClass.RT	1
RtClass.IRT	2

L'énumération SyncRole a les valeurs suivantes.

Valeur d'énumération	Valeur numérique
SyncRole.NotSynchronized	0
SyncRole.SyncMaster	1
SyncRole.SyncSlave	2
SyncRole.RedundantSyncMaster	4

Attributs pour le numéro d'appareil

Les attributs pour le numéro d'appareil sont indiqués ci-dessous.

Nom d'attribut	Type de données	Type	Accès	Description
PnDeviceNumber	Int	Lecture/écriture	Attribut dynamique	Indique le numéro d'appareil

Code de programme : appeler et définir les attributs de l'IoConnector

```
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);
```

Voir aussi

Ouvrir un projet (Page 99)

7.16.12 Accéder à un contrôleur d'adresse

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Un élément d'appareil qui est en même temps contrôleur d'adresse offre des fonctions supplémentaires. Pour accéder aux adresses enregistrées du contrôleur d'adresse, il faut utiliser le rôle AddressController.

Code de programme : appeler un contrôleur d'adresse

Pour appeler le rôle de contrôleur d'adresse, modifiez le code de programme suivant :

```
AddressController addressController =  
(IEngineeringServiceProvider)deviceItem).GetService<AddressController>();  
if (addressController != null)  
{  
    ... // work with the address controller  
}
```

Attributs d'un contrôleur d'adresse

Un contrôleur d'adresse possède les attributs suivants :

- RegisteredAddresses

Pour appeler les attributs d'un contrôleur d'adresse, modifiez le code de programme suivant :

```
AddressController addressController = ...;  
foreach (Address registeredAddress in addressController.RegisteredAddresses)  
{  
    ...  
}
```

7.16.13 Accéder à des adresses

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les objets adresse sont appelés à l'aide du lien de composition `Addresses` d'un élément d'appareil. L'attribut `Addresses` retourne un recueil de `AddressComposition` qui peut être énuméré.

Code de programme : appeler l'adresse d'un élément d'appareil

Pour appeler l'adresse d'un élément d'appareil, modifiez le code de programme suivant :

```
AddressComposition addresses = deviceItem.Addresses;
foreach(Address address in addresses)
{
    // work with the address
}
```

Code de programme : appeler l'adresse d'un contrôleur IO

Pour appeler l'adresse d'un contrôleur IO, modifiez le code de programme suivant :

```
AddressComposition addresses = ioController.Addresses;
foreach(Address address in addresses)
{
    // work with the address
}
```

Attributs

L'adresse prend en charge les attributs suivants :

Nom d'attribut	Type de données	Accessible en écriture	Accès	Commentaire
AddressControllers	AddressControllerAssociation	read		
Context	enum: AddressContext	read	Dynamique	Uniquement pour adresses de diagnostic et pour éléments d'appareil spécifiques
IoType	enum: AddressIoType	read		

Nom d'attribut	Type de données	Accessible en écriture	Accès	Commentaire
StartAddress	Int32	read/write		
Length	Int32	read		

Valeur	Description
AddressIoType.Diagnosis	Le type IO d'adresse est diagnostic.
AddressIoType.Input	Le type IO d'adresse est entrée.
AddressIoType.Output	Le type IO d'adresse est sortie.
AddressIoType.Substitute	Le type IO d'adresse est remplacement.
AddressIoType.None	Le type IO d'adresse n'est pas indiqué.

Valeur	Description
AddressContext.None	Le contexte d'adresse n'est pas valable.
AddressContext.Device	Contexte d'adresse d'appareil.
AddressContext.Head	Contexte d'adresse de tête.

Code de programme : Lire les attributs

Pour appeler les attributs, modifiez le code de programme suivant :

```
AddressControllerAssociation addressControllers = address.AddressControllers;
Int32 startAddress = address.StartAddress;
AddressIoType addressType = address.IoType;
Int32 adressLength = address.Length;
```

Code de programme : écrire les attributs

Pour écrire les attributs, modifiez le code de programme suivant :

```
Address addressControllers = ...;
address.StartAddress = intValueStartAddress;
```

Code de programme : attributs pour accès dynamique

Pour appeler les attributs, modifiez le code de programme suivant :

```
Address address= ...;
object attributeValue = ((IEngineeringObject)address).GetAttribute("Context");
```


7.16.14 Accéder à l'identifiant de matériel

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les objets "Identifiant de matériel" sont appelés par les objets suivants :

- Device
- DeviceItem
- IoSystem

L'identifiant de matériel est représenté par la classe `HwIdentifier` et appelé avec l'attribut `HwIdentifiers`.

Code de programme : appeler l'identifiant de matériel

Pour mettre à disposition `HwIdentifier`, modifiez le code de programme suivant :

```
var hwObject = ...
foreach(HwIdentifier hardwareIdentifier in hwObject.HwIdentifiers)
{
    // Work with the HwIdentifier
}
```

Attributs d'un identifiant de matériel

```
HwIdentifierControllerAssociation controllers = hwIdentifier.HwIdentifierControllers;
Int64 Identifier = hwIdentifier.Identifier;
```

7.16.15 Accéder au contrôleur d'identifiant de matériel

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Quand un élément d'appareil est en même temps contrôleur d'identifiant de matériel, il est possible d'accéder aux identifiants de matériel enregistrés. Pour accéder à ces HwIdentifierController, il faut utiliser un certain service de l'élément d'appareil.

Code de programme : appeler le contrôleur d'identifiant de matériel

Pour appeler le HwIdentifierController, modifiez le code de programme suivant :

```
HwIdentifierController hwIdentifierController =  
((IEngineeringServiceProvider)deviceItem).GetService<HwIdentifierController>();  
if (hwIdentifierController != null)  
{  
    ... // work with the hardware identifier controller  
}
```

Code de programme : attributs d'un contrôleur d'identifiant de matériel

Un contrôleur d'adresse possède les attributs suivants :

- RegisteredHwIdentifiers : les contrôleurs d'identifiant de matériel sur lesquels l'identifiant de matériel est enregistré.

Pour appeler les attributs d'un contrôleur d'adresse, modifiez le code de programme suivant :

```
HwIdentifierController hwIdentifierController = ...;  
HwIdentifierAssociation controllers = hwIdentifierController.RegisteredHwIdentifiers;
```

7.16.16 Accéder aux voies d'éléments d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Une voie est représentée par la classe Channel. Les voies sont appelées par un élément d'appareil au moyen de l'attribut Channels de la classe DeviceItem. L'attribut Channels retourne une implémentation de ChannelComposition qui peut être énumérée. Quand l'élément d'appareil n'a pas de voie, l'attribut Channels retourne un recueil vide.

Attributs obligatoires

Une voie prend en charge les attributs obligatoires suivants :

Nom d'attribut	Type de données	Accessible en écriture	Accès	Commentaire
IoType	ChannelIoType	read		
Type	ChannelType	read		
Number	Int32	read		
ChannelAddress	Int32	read	Dynamique	Adresse de la voie en bits
ChannelWidth	UInt32	read	Dynamique	Largeur de la voie en bits

Code de programme : appeler les voies d'un élément d'appareil

Pour appeler les voies d'un élément d'appareil, modifiez le code de programme suivant :

```
ChannelComposition channels = deviceItem.Channels
foreach(Channel channel in channels)
{
    // work with the channel
}
```

Code de programme : attributs obligatoires d'une voie

Pour appeler les voies d'un élément d'appareil, modifiez le code de programme suivant :

```
Channel channel = ...;
int channelNumber = channel.Number;
ChannelType type = channel.Type;
ChannelIoType ioType = channel.IoType;
```

Code de programme : appeler les valeurs des attributs pour l'accès dynamique

Pour appeler les valeurs d'attributs dynamiques, modifiez le code de programme suivant :

```
Channel channel = ...;
Int32 channelAddress = (Int32)((IEngineeringObject)channel).GetAttribute("ChannelAddress");
UInt32 channelWidth = (UInt32)((IEngineeringObject)channel).GetAttribute("ChannelWidth");
```

Code de programme : déterminer la valeur d'un attribut dynamique

Pour déterminer la valeur d'un attribut dynamique accessible en écriture, modifiez le code de programme suivant :

```
Channel channel = ...;
((IEngineeringObject)channel).SetAttribute("AnAttribute", 1234);
```

7.17 Fonctions sur les données d'un appareil HMI

7.17.1 Vues

7.17.1.1 Créer des dossiers de vues personnalisés

Condition

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour créer un dossier de vues personnalisé, modifiez le code de programme suivant :

```
//Creates a screen folder
private static void CreateScreenFolder(HmiTarget hmitarget)
{
    ScreenUserFolder myCreatedFolder =
hmitarget.ScreenFolder.Folders.Create("myScreenFolder");
}
```

7.17.1.2 Supprimer la vue d'un dossier

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Remarque

Vous ne pouvez pas supprimer une fenêtre permanente. Une fenêtre permanente est une vue système toujours existante.

Code du programme

Pour supprimer une vue d'un certain dossier, modifiez le code de programme suivant :

```
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();
    }
}
```

7.17.1.3 Supprimer un modèle de vue d'un dossier

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un appareil IHM existe dans le projet.

Code du programme

Pour supprimer un modèle de vue d'un certain dossier, modifiez le code de programme suivant :

```
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();
    }
}
```

7.17.1.4 Supprimer toutes les vues d'un dossier

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Remarque

Vous ne pouvez pas supprimer une fenêtre permanente. Une fenêtre permanente est une vue système toujours existante.

Code du programme

Pour supprimer toutes les vues d'un certain dossier, modifiez le code de programme suivant :

```
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();
    }
}
```

7.17.2 Cycles

7.17.2.1 Suppression de cycle

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un appareil IHM existe dans le projet.

Utilisation

Vous ne pouvez pas supprimer les cycles prédéfinis.

À l'aide de la composition dans le modèle objet (composition count) du cycle concerné, vous pouvez déterminer si des cycles ont effectivement été supprimés. Il n'est plus possible d'accéder à ces cycles.

Code du programme

Pour supprimer un cycle d'un appareil IHM, modifiez le code de programme suivant :

```
public static void DeleteCycle(HmiTarget hmiTarget)
{
    CycleComposition cycles = hmiTarget.Cycles;
    Cycle cycle = cycles.Find("myCycle");
    cycle.Delete();
}
```

7.17.3 Listes de textes

7.17.3.1 Suppression de la liste de textes

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un appareil IHM existe dans le projet.

Code du programme

Pour supprimer une liste de textes sélectionnée et toutes les entrées de liste correspondantes d'un appareil IHM, modifiez le code de programme suivant :

```
public static void DeleteTextList(HmiTarget hmiTarget)
{
    TextListComposition textLists = hmiTarget.TextLists;
    TextList textList = textLists.Find("myTextList");
    textList.Delete();
}
```

7.17.4 Listes de graphiques

7.17.4.1 Suppression d'une liste de graphiques

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un appareil IHM existe dans le projet.

Code du programme

Pour supprimer une liste de graphiques sélectionnée et toutes les entrées de liste correspondantes d'un appareil IHM, modifiez le code de programme suivant :

```
private static void DeleteGraphicList(HmiTarget hmiTarget)
{
    GraphicListComposition graphicLists = hmiTarget.GraphicLists;
    GraphicList graphicList = graphicLists.Find("myGraphicList");
    graphicList.Delete();
}
```


7.17.5 Connexions

7.17.5.1 Suppression de la liaison

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un appareil IHM existe dans le projet.

Code du programme

Pour supprimer une liaison de communication sélectionnée d'un appareil IHM, modifiez le code de programme suivant :

```
private static void DeleteConnection(HmiTarget hmiTarget)
{
    ConnectionComposition connections = hmiTarget.Connections;
    Connection connection = connections.Find("HMI_connection_1");
    connection.Delete();
}
```

7.17.6 Table des variables

7.17.6.1 Générer des dossiers personnalisés pour variables IHM

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour créer un dossier personnalisé pour variables HMI, modifiez le code de programme suivant :

```
private static void CreateUserFolderForHMITags(HmiTarget hmitarget)
// Creates an HMI tag user folder
{
    TagSystemFolder folder = hmitarget.TagFolder;
    TagUserFolder myCreatedFolder = folder.Folders.Create("MySubFolder");
}
```

7.17.6.2 Enumérer les variables d'une table de variables IHM

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour énumérer toutes les variables d'une table de variables IHM, modifiez le code de programme suivant :

```
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
    }
}
```

7.17.6.3 Suppression de variables individuelles d'une table de variables IHM

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour supprimer une variable déterminée d'une table des variables IHM, modifiez le code de programme suivant :

```
private static void DeleteATag(HmiTarget hmiTarget)
{
    string tagName = "MyTag";
    TagTable defaultTagTable = hmiTarget.TagFolder.DefaultTagTable;
    TagComposition Variablen = defaultTagTable.Tags;
    Tag tag = tags.Find(tagName);
    tag.Delete();
}
```

7.17.6.4 Supprimer une table de variables d'un dossier

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un appareil IHM existe dans le projet.

Utilisation

Vous ne pouvez pas supprimer la table des variables standard.

Code du programme

Modifiez le code de programme suivant :

```
// 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();
}
```

7.17.7 Scripts VB

7.17.7.1 Créer des dossiers personnalisés pour les scripts

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour créer un sous-dossier personnalisé pour scripts dans un dossier système ou un autre dossier personnalisé, modifiez le code de programme suivant :

```
private static void CreateFolderInScriptfolder(HmiTarget hmitarget)
//Creates a script user subfolderVBScriptSystemFolder
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptUserFolderComposition vbScriptFolders = vbScriptFolder.Folders;
    VBScriptUserFolder vbScriptSubFolder = vbScriptFolders.Create("mySubfolder");
}
```

7.17.7.2 Supprimer les scripts VB d'un dossier

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un appareil IHM existe dans le projet.

Code du programme

Pour supprimer un script VB d'un certain dossier, modifiez le code de programme suivant :

```
//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();
    }
}
```

7.17.8 Supprimer le dossier personnalisé d'un pupitre opérateur

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code de programme

Pour supprimer un dossier personnalisé d'un pupitre opérateur, modifiez le code de programme suivant :

```
HmiTarget hmiTarget = ...;
ScreenUserFolder screenUserGroup = hmiTarget.ScreenFolder.Folders.Find("MyUserFolder");
screenUserGroup.Delete();
```

7.18 Fonctions sur les données d'un appareil API

7.18.1 Déterminer le statut d'un API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)

Utilisation

Vous pouvez déterminer l'état d'un API ou de tous les API au sein d'un projet.

TIA Portal Openness distingue les états suivants :

- Offline
- l'API est connecté ("la connexion est établie")
- En ligne
- l'API n'est pas connecté ("la connexion est coupée")
- Incompatible
- accès impossible
- Protégé

Code du programme

Pour déterminer l'état d'un API, modifiez le code de programme suivant :

```
public static OnlineState GetOnlineState(DeviceItem deviceItem)
{
    OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
    return onlineProvider.State;
}
```

Pour déterminer l'état de tous les API dans un projet, modifiez le code de programme suivant :

```
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;
            }
        }
    }
}
```

7.18.2 Accéder aux paramètres d'une liaison en ligne

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'interface TIA Portal Openness API vous permet de définir les paramètres pour une liaison en ligne :

- Enumérer les types de connexion disponibles avec un API
- Enumérer les interfaces disponibles avec un API
- Enumérer les emplacements affectés
- Enumérer les adresses disponibles des sous-réseaux et passerelles
- Définir les paramètres de liaison

Code de programme : déterminer les paramètres de liaison

Pour énumérer les modes de liaison, interfaces de PC et emplacements disponibles, modifiez le code de programme suivant :

```
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);
            }
        }
    }
}
```

Vous pouvez aussi accéder à un type de connexion et une interface de PC par le nom :

```
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;
}
```


Pour énumérer les adresses disponibles des sous-réseaux et passerelles sur une interface de PC, modifiez le code de programme suivant :

```
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}", gatewayAddress.Name);
            }
        }
    }
}
```

Vous pouvez également accéder aux sous-réseaux et passerelles par le nom ou l'adresse IP :

```
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");
}
```

Code de programme : Définir les paramètres de liaison

Remarque

La définition des paramètres de liaison écrase tous les paramètres de liaison définis auparavant. Si vous avez déjà défini les paramètres de liaison directement dans le portail TIA, vous n'avez pas besoin d'appeler `ApplyConfiguration`. Si une liaison en ligne existe déjà vers un API pendant l'appel de `ApplyConfiguration`, une exception sera déclenchée.

Pour définir les paramètres d'emplacement, modifiez le code de programme suivant :

```
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();
}
```

Pour définir les paramètres d'adresse de passerelles, modifiez le code de programme suivant :

```
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();
}
```

Pour définir les paramètres d'adresse de sous-réseaux, modifiez le code de programme suivant :

```
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();
}
```

7.18.3 Fonctions de chargement de données dans l'API

7.18.3.1 Charger des composants matériels et logiciels dans l'API

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)

Utilisation

L'utilisateur d'Openness peut charger des composants logiciels et matériels dans l'API via DownloadProvider (accès via le DeviceItem). Si un DeviceItem représente une cible pouvant être chargée, une instance de DownloadProvider est fournie en retour en cas d'appel de GetService. Sinon, le service fournit null en retour.

Code de programme : appeler le service DownloadProvider depuis un élément d'appareil

```
DeviceItem deviceItem = ...;
DownloadProvider downloadProvider = deviceItem.GetService<DownloadProvider>();
if (downloadProvider != null)
{
    ...
}
```

Paramètres de la méthode Download

Pour charger des données dans un API, l'utilisateur appelle la méthode Download de DownloadProvider. La méthode Download comprend quatre paramètres : l'objet IConfiguration, deux délégués et DownloadOptions (matériel, logiciel ou matériel et logiciel).

Nom de paramètre	Type	Description
configuration	Siemens.Engineering.Connection.IConfiguration	Configuration de la connexion à un appareil.
preDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Délégué à appeler pour le contrôle de la configuration avant la procédure de chargement.

Nom de paramètre	Type	Description
postDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Délégué à appeler pour le contrôle de la configuration après la procédure de chargement.
downloadOptions	Siemens.Engineering.Download.DownloadOptions	Options de chargement.

- La procédure de chargement Openness n'est prise en charge que si les configurations sont réalisées correctement par l'utilisateur. Si la configuration n'est pas valide, une `EngineeringTargetInvocationException` est émise et la procédure de chargement est annulée. Les API activés avec sécurité ne sont pas pris en charge pour la procédure de chargement.
- La compilation faisant partie de la procédure de chargement, il est recommandé d'exécuter la compilation avant la procédure de chargement afin d'analyser les résultats de la compilation.
- Openness ne prend en charge que l'option de chargement complet.

Paramètre 1 : IConfiguration

L'utilisateur doit indiquer l'objet `IConfiguration` comme premier paramètre de la méthode `Download`. Il sert à établir une connexion à l'API spécifié. L'interface `IConfiguration` est mise en œuvre par `ConfigurationAddress` et `ConfigurationTargetInterface`. Il est possible d'accéder aux deux objets via l'instance `ConnectionConfiguration`. L'instance `ConnectionConfiguration` peut être déterminée via la propriété `DownloadProvider.Connection: ConnectionConfiguration` ou en option via `OnlineProvider.Connection: ConnectionConfiguration`.

La configuration de l'objet `ConnectionConfiguration` est décrite à la rubrique [Accéder aux paramètres d'une liaison en ligne \(Page 255\)](#).

```
...
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];
...
```

Paramètres 2 et 3 : DownloadConfigurationDelegate

L'utilisateur d'Openness doit mettre à disposition deux mises en œuvre de `DownloadConfigurationDelegate(DownloadConfiguration downloadConfiguration)` vides. Le premier délégué est appelé pour les configurations avant la procédure de chargement et le second une fois la procédure de chargement terminée. Les délégués sont appelés pour chaque configuration pour laquelle une action de l'utilisateur est nécessaire. Pour plus d'informations sur les rappels, référez-vous à [Prise en charge de rappels \(Page 270\)](#) Certaines configurations ne comprennent qu'une seule information et ne nécessitent alors pas d'action de l'utilisateur.

Les types possibles de configuration de chargement sont décrits ci-après.

Nom de configuration	Description et propriétés
DownloadConfiguration	<ul style="list-style-type: none"> • Classe de base pour toutes les configurations. • Contient l'unique propriété DownloadConfiguration.Message : string (propriété protégée en écriture contenant le message de configuration)
DownloadSelectionConfiguration	<ul style="list-style-type: none"> • Classe de base pour toutes les configurations pouvant être sélectionnées. • Ne contient aucune autre propriété. Une sélection doit être mise à disposition dans toutes les classes subordonnées dérivées de cette classe.
DownloadCheckConfiguration	<ul style="list-style-type: none"> • Classe de base pour toutes les configurations contrôlées ou non. • Contient l'unique propriété DownloadCheckConfiguration.Checked: bool<string> (propriété accessible en lecture/écriture indiquant si la configuration est contrôlée ou non)
DownloadPasswordConfiguration	<ul style="list-style-type: none"> • Classe de base pour toutes les configurations nécessitant un mot de passe pour la procédure de chargement. • Contient une méthode unique de définition du mot de passe. DownloadPasswordConfiguration.SetPassword (password: SecureString) : void

Le type de données des configurations est expliqué ci-après.

Configuration	Type de données	Description et action
DownloadSelection-Configuration	StartModules	Set CurrentSelection:StopModulesSelections. Les valeurs d'énumération disponibles sont NoAction (No action) StopAll (Stop all) Ces modules sont arrêtés pour un chargement dans un appareil.
	StopModules	Set CurrentSelection:StartModulesSelections. Valeurs d'énumération disponibles : NoAction (No action) StartModule (Start module) Ces modules sont démarrés une fois la procédure de chargement terminée.
	AllBlocksDownload	Set CurrentSelection:AllBlocksDownloadSelections. Valeur d'énumération disponible : DownloadAllBlocks (Download all blocks to the device) Charge le logiciel dans l'appareil.
	OverwriteSystemData	Set CurrentSelection:OverwriteSystemDataSelections. Les valeurs d'énumération disponibles sont NoAction (No action) Overwrite (Download to device) Supprime et écrase les données système disponibles dans le lieu de stockage cible.
	ConsistentBlocksDownload	Set CurrentSelection:ConsistentBlocksDownloadSelections. Valeur d'énumération disponible : ConsistentDownload (Consistent download) Charge le logiciel dans l'appareil.
	AlarmTextLibrariesDownload	Set CurrentSelection:AlarmTextLibrariesDownloadSelections. Les valeurs d'énumération disponibles sont ConsistentDownload (Consistent download) NoAction (No action) Charge tous les textes d'alarme et toutes les listes de texte.
	ProtectionLevelChanged	Set CurrentSelection:ProtectionLevelChangedSelections. Les valeurs d'énumération disponibles sont NoChange (No change) ContinueDownloading (Continue downloading to the device) La protection de la CPU est définie sur le prochain niveau plus bas.
	ActiveTestCanBeAborted	Set CurrentSelection:ActiveTestCanBeAbortedSelections. Les valeurs d'énumération disponibles sont NoAction (No action) AcceptAll (Accept all) Les fonctions de test et de mise en service actives sont annulées pendant la procédure de chargement dans l'appareil.

Configuration	Type de données	Description et action
	ResetModule	Les valeurs d'énumération Set CurrentSelection:ResetModuleSelections disponibles sont NoAction (No action) DeleteAll (Delete all) Réinitialise le module.
	LoadIdentificationData	Set CurrentSelection:LoadIdentificationDataSelections. Les valeurs d'énumération disponibles sont LoadNothing (Load nothing) LoadData (Load data) LoadSelected (Load selected) Charger les données d'identification dans les appareils PROFINET IO et leurs modules.
	DifferentTargetConfiguration	Set CurrentSelection:DifferentTargetConfigurationSelections. Les valeurs d'énumération disponibles sont NoAction (No action) AcceptAll (Accept all) Indique la différence entre le module configuré et le module cible (en ligne).
	InitializeMemory	Set CurrentSelection:InitializeMemorySelections. Valeurs d'énumération disponibles : NoAction (No action) AcceptAll (Accept all) Ce type de données sert à l'initialisation de la mémoire.
DownloadCheckConfiguration	CheckBeforeDownload	Propriété Set IsChecked:bool Effectue un contrôle avant le chargement dans l'appareil.
	UpgradeTargetDevice	Propriété Set IsChecked:bool Contrôle les différentes versions de projet dans l'appareil configuré et dans l'appareil cible (en ligne).
	OverWriteHMIData	Propriété Set IsChecked:bool Écrase les objets en ligne.
	FitHMIComponents	Propriété Set IsChecked:bool Les composants avec une autre version sont installés sur l'appareil cible.
	TurnOffSequence	Propriété Set IsChecked:bool Désactive la séquence avant le chargement.
	OverwriteTargetLanguages	Propriété Set IsChecked:bool Pour différencier les paramètres du projet et de la programmation de l'API
	DowngradeTargetDevice	Propriété Set IsChecked:bool Pour mentionner les différents formats de fichier dans des projets en ligne et hors ligne.

Configuration	Type de données	Description et action
DownloadPassword-Configuration	ModuleReadAccessPassword	Méthode Set password via SetPassword(password:SecureString) Saisir le mot de passe afin d'obtenir un accès en lecture au module.
	ModuleWriteAccessPassword	Méthode Set password via SetPassword(password:SecureString) Saisir le mot de passe afin d'obtenir un accès en écriture au module.
	BlockBindingPassword	Méthode Set password via SetPassword(password:SecureString) Méthode pour la configuration d'un mot de passe lié à un bloc.

IMPORTANT

Tenez compte du fait que les configurations de chargement sont semblables aux configurations dans les dialogues avec aperçu du chargement et ressemblent aux résultats de chargement si vous travaillez avec l'interface utilisateur de TIA Portal.

**ATTENTION**

L'utilisateur de l'API est responsable des mesures de sécurité pour la gestion des mots de passe par code.

Une configuration non traitée pouvant empêcher la procédure de chargement entraîne une EngineeringTargetInvocationException et annule la procédure de chargement. Une

EngineeringDelegateInvocationException est émise en cas d'exception non traitée dans le délégué.

Exemple de mise en œuvre de PreDownloadDelegate :

```
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 delagate will cancel
download
}
```

Exemple de mise en œuvre de PostDownloadDelegate :

```
private static void PostConfigureDownload(DownloadConfiguration downloadConfiguration)
{
    StartModules startModules = downloadConfiguration as StartModules;
    if (startModules != null)
    {
        startModules.CurrentSelection = StartModulesSelections.StartModule; //
Sets PLC in "Run" mode
    }
}
```

Paramètre 4 : DownloadOptions

L'utilisateur doit indiquer les options de chargement (marquées par enum) via DownloadOptions. Ce paramètre détermine le type de procédure de chargement à exécuter, par exemple matériel, logiciel ou matériel et logiciel.

```
[Flags]
public enum DownloadOptions {
    None = 0, // Download nothing
    Hardware, // Download hardware only
    Software // Download software only
}
```

Si l'utilisateur souhaite charger le matériel et le logiciel dans l'appareil, DownloadOptions.Hardware | DownloadOptions.Software doit être indiqué comme 4ème paramètre de la méthode Download.

DownloadResult

Le DownloadResult obtenu par l'action Download fournit l'état des objets chargés en retour.

Exemple d'appel de la méthode Download

```
[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);
    }
}
}
```

7.18.3.2 Démarrage et arrêt d'un API

Conditions

- L'application Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

En cas d'interactions avec TIA Portal via l'API Openness, il peut être nécessaire de changer le mode de fonctionnement de l'automate. TIA Openness offre la possibilité de modifier l'état de fonctionnement de l'API soit sur Start, soit sur Stop.

Code du programme

Pour régler l'état de fonctionnement de l'API sur STOP, modifiez le code de programme suivant.

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
{
    StopModules stopModules = downloadConfiguration as StopModules;
    if (stopModules != null)
    {
        // Puts PLC in "Stop" mode
        stopModules.CurrentSelection = StopModulesSelections.StopAll;
    }
}
```

Pour régler l'état de fonctionnement de l'API sur START, modifiez le code de programme suivant.

```
public void ConfigurePostDownload(DownloadConfiguration downloadConfiguration)
{
    StartModules startModules = downloadConfiguration as StartModules;
    if (startModules != null)
    {
        // Puts PLC in "Start" mode
        startModules.CurrentSelection = StartModulesSelections.StartModule;
    }
}
```

7.18.3.3 Prise en charge de rappels

Conditions

- L'application Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Au cours de leur exécution, certaines méthodes de l'API nécessitent une interaction avec le code d'application défini par l'utilisateur. Les Delegates servent à gérer ces actions de rappel dans le code d'application personnalisé. Vous devez mettre en œuvre une méthode avec une signature compatible et la transmettre à l'action en tant que paramètre delegate. Le TIA Portal appelle les méthodes mises en œuvre pour l'exécution.

Code de programme

```
// This delegate is declared in Siemens.Engineering.dll
public delegate void
Siemens.Engineering.Download.DownloadConfigurationDelegate (Siemens.Engineering.Download.Co
nfigurations.DownloadConfiguration configuration);
...
```

Exemple de code d'application personnalisé utilisant et mettant en œuvre 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
}
```

Remarque

L'attribut **STAThread** assure que les délégués (Delegates) sont appelés dans le thread principal de l'exécution.

7.18.3.4 Protection par mot de passe du PLC

Conditions

- L'application Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)
- L'API est hors ligne.

Utilisation

Dans les interactions avec TIA Portal via Openness API, il peut être nécessaire de changer le niveau de protection de l'automate. TIA Openness offre la possibilité de protéger l'API par un mot de passe. Le mot de passe peut être aussi bien paramétré pour des API en écriture seule que lecture seule.

Code de programme

Pour les API en écriture seule, modifiez le code de programme suivant :

```
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
    }
}
```

Pour les API en lecture seule, modifiez le code de programme suivant :

```
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
    }
}
```



ATTENTION

L'utilisateur de l'API est responsable des mesures de sécurité pour la gestion des mots de passe par code.

7.18.3.5 Manipulation des mots de passe d'API liée aux blocs

Conditions

- L'application Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

TIA Openness prend en charge la liaison de données de mots de passe pour les applications client. TIA Openness donne au client un moyen de spécifier un mot de passe lié au bloc. Par exemple, un mot de passe lié au bloc peut être configuré à la catégorie DownloadPasswordConfiguration en faisant appel à la méthode SetPassword.

Remarque

Si vous souhaitez protéger la procédure de chargement avec un password, un password doit être saisi à chaque appel de la fonction Download. Cela s'applique toujours, que l'appareil soit configuré ou non. Après la saisie correcte du mot de passe pour une configuration donnée, tous les appels suivants de SetPassword sont ignorés.

Code de programme

Modifiez le code de programme suivant :

```
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);
    }
}
```

7.18.4 Comparer le logiciel de l'API

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet avec votre application TIA Portal Openness.
Voir Ouvrir un projet (Page 99)

Utilisation

Pour déterminer la divergence entre les logiciels de deux appareils, vous disposez des possibilités suivantes :

- Comparaison entre les logiciels de deux API configurés
- Comparaison entre le logiciel d'un API et la bibliothèque de projet
- Comparaison entre le logiciel d'un API et la bibliothèque globale
- Comparaison entre le logiciel d'un API et la copie maître d'un API
- Comparaison entre le logiciel d'un API configuré et le logiciel d'un API connecté à l'état "En ligne"

Signature

Utilisez la méthode `CompareTo` ou `CompareToOnline` pour la comparaison.

```
public CompareResult CompareTo (ISoftwareCompareTarget compareTarget)
public CompareResult CompareToOnline ()
```

Valeur de retour/paramètre	Fonction
<code>CompareResult compareResult</code>	Fournit en retour le résultat de comparaison : <ul style="list-style-type: none"> • <code>FolderContentsDifferent</code> : le contenu des dossiers comparés diffère. • <code>FolderContentsIdentical</code> : le contenu des dossiers comparés est identique. • <code>ObjectsDifferent</code> : le contenu des objets comparés diffère. • <code>ObjectsIdentical</code> : le contenu des objets comparés est identique. • <code>LeftMissing</code> : l'objet n'est pas compris dans l'objet à partir duquel la comparaison a été lancée. • <code>RightMissing</code> : l'objet n'est pas compris dans l'objet auquel la comparaison s'applique.
<code>ISoftwareCompareTarget compareTarget</code>	Liste d'objets comparables.

Code de programme

Pour afficher le résultat de la comparaison, modifiez le code de programme suivant :

```
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);
    }
}
```

Pour comparer les logiciels des appareils, modifiez le code de programme suivant :

```
private static void CompareTwoOfflinePlcs(PlcSoftware plcSoftware0, PlcSoftware
plcSoftware1)
{
    if (plcSoftware0 != null && plcSoftware1 != null)
    {
        CompareResult compareResult = plcSoftware0.CompareTo(plcSoftware1);
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

Pour comparer le logiciel d'un API avec la bibliothèque de projet, modifiez le code de programme suivant :

```
private static void ComparePlcToProjectLibrary(Project project, PlcSoftware plcSoftware)
{
    if (project != null && plcSoftware != null)
    {
        CompareResult compareResult = plcSoftware.CompareTo(project.ProjectLibrary);
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

Pour comparer le logiciel d'un API avec la bibliothèque globale, modifiez le code de programme suivant :

```
private static void ComparePlcToGlobalLibrary(PlcSoftware plcSoftware, GlobalLibrary
globalLibrary)
{
    if (plcSoftware != null && globalLibrary != null)
    {
        CompareResult compareResult = plcSoftware.CompareTo(globalLibrary);
        WriteResult(compareResult.RootElement, String.Empty);
    }
}
```

Pour comparer le logiciel d'un API avec une copie maître, modifiez le code de programme suivant :

```
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);
    }
}
```

Pour comparer le logiciel d'un API avec le logiciel d'un API connecté, modifiez le code de programme suivant :

```
private static void ComparePlcToOnlinePlc(PlcSoftware plcSoftware)
{
    if (plcSoftware != null)
    {
        CompareResult compareResult = plcSoftware.CompareToOnline();
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

7.18.5 Etablir ou interrompre une liaison en ligne à l'API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Tous les appareils sont énumérés.
Voir Appeler des éléments d'appareil (Page 227).

Utilisation

Vous pouvez établir la liaison en ligne à un API ou interrompre une liaison en ligne existante.

Code du programme

Pour établir ou interrompre la liaison en ligne à un API, modifiez le code de programme suivant :

```
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();
}
```

Vous pouvez également établir ou interrompre les liaisons en ligne à tous les API disponibles dans un projet.

```
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();
            }
        }
    }
}
```

7.18.6 Blocs

7.18.6.1 Interroger le groupe "Blocs de programme"

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un API est décelé dans le projet.

Code du programme

Pour interroger le groupe "Blocs de programme", modifiez le code de programme suivant :

```
private static void GetBlockGroupOfPLC(PlcSoftware plcsoftware)
//Retrieves the system group of a block
{
    PlcBlockSystemGroup blockGroup = plcsoftware.BlockGroup;
}
```

7.18.6.2 Enumérer les groupes Blocs personnalisés

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un API est décelé dans le projet.

Utilisation

Les sous-groupes compris sont considérés comme récurrents lors de l'énumération.

Code du programme : Enumérer tous les groupes

Pour énumérer les groupes Blocs personnalisés, modifiez le code de programme suivant :

```
//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
    }
}
```

Code du programme : Accéder à un groupe

Pour accéder à un groupe de blocs personnalisé sélectionné, modifiez le code de programme suivant :

```
//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");
}
```

7.18.6.3 Enumérer tous les blocs

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Un API est décelé dans le projet.

Utilisation

Il est possible d'accéder de manière ciblée à un bloc de programme si son nom est connu.

Code du programme : Enumérer tous les blocs

Pour énumérer les blocs de tous les groupes Blocs, modifiez le code de programme suivant :

```
private static void EnumerateAllBlocks(PlcSoftware plcsoftware)
//Enumerates all blocks
{
    foreach (PlcBlock block in plcsoftware.BlockGroup.Blocks)
    {
        // Do something...
    }
}
```

Code du programme : Accéder à un bloc déterminé

Pour accéder à un bloc donné, modifiez le code de programme suivant :

```
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");
}
```

7.18.6.4 Interroger les informations d'un bloc/type de données utilisateur

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

La TIA Portal Openness API prend en charge l'interrogation des informations suivantes pour le programme et les blocs de données et pour les types de données utilisateur :

- Horodatage au format UTC
L'horodatage fournit les informations suivantes :
 - Quand le bloc a-t-il été compilé pour la dernière fois ?
 - Quand le bloc a-t-il été modifié pour la dernière fois ?
- Attribut "Consistency"
L'attribut « Consistency » est mis sur "True" dans les cas suivants :
 - Le bloc a été correctement compilé.
 - Le bloc n'a pas été modifié depuis la compilation.
 - Aucune modification ayant impliqué une nouvelle compilation n'a été apportée aux objets externes.
- Langage de programmation utilisé (uniquement programme et de blocs de données)
- Numéro de bloc
- Nom de bloc
- Auteur du bloc
- Famille de bloc
- Titre du bloc
- Version du bloc

Pour plus d'informations, voir Blocs et types de modèle d'objet TIA Portal Openness (Page 56).

Code du programme

Pour interroger les informations susmentionnées, modifiez le code de programme suivant :

```
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;
}
```

Voir aussi

Importation de données de configuration (Page 377)

7.18.6.5 Supprimer un bloc

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Code du programme

Pour supprimer un bloc, modifiez le code de programme suivant :

```
//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();
        }
    }
}
```

Voir aussi

Importation de données de configuration (Page 377)

7.18.6.6 Créer un groupe pour blocs

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour créer un groupe pour blocs, modifiez le code de programme suivant :

```
private static void CreateBlockGroup(PlcSoftware plcsoftware)
//Creates a block group
{
    PlcBlockSystemGroup systemGroup = plcsoftware.BlockGroup;
    PlcBlockUserGroupComposition groupComposition = systemGroup.Groups;
    PlcBlockUserGroup myCreatedGroup = groupComposition.Create("MySubGroupName");
}
```

Voir aussi

Importation de données de configuration (Page 377)

7.18.6.7 Accéder aux attributs de tous les blocs

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'utilisateur peut définir les attributs valables pour tous les blocs à l'aide de la méthode `SetAttribute()`. Les exemples de code suivants sont basés sur les deux attributs `AutoNumber` et `Number` (voir Exporter des blocs (Page 448) pour tous les attributs de blocs valables).

Code de programme :

```
...
PlcBlockGroup blockFolder = YourUtilities.GetFolder();
var block = blockFolder.Blocks.Find("Block_1");
if (block.GetAttribute("AutoNumber")==true)
{
    block.SetAttribute("AutoNumber", false);
}
block.SetAttribute("Number", 2);
...
```

7.18.6.8 Créer un FB ProDiag

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'utilisateur d'Openness peut créer un FB ProDiag via l'action Create de la combinaison PLCBlock avec les paramètres suivants.

1. Nom
 2. Mémento de numérotation automatique
 3. Numéro (si le memento de numérotation automatique est vrai)
 4. Langage de programmation
- Si l'utilisateur appelle l'action Create avec le langage de programmation ProDiag, un nouveau FB est créé sans IDB.
 - Si l'utilisateur appelle l'action Create avec un IDB de ProDiag, l'IDB est créé par ProDiag.
 - Dans tous les autres cas non pris en charge, une exception récupérable est déclenchée.

Code de programme : Créer un FB ProDiag

```
...
PlcBlockGroup blockFolder = YourUtilities.GetFolder();
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;ProgrammingLanguage iDBLanguage =
ProgrammingLanguage.DB;InstanceDB iDbBlock = blockComposition.CreateInstanceDB(iDBName,
isAutoNumber, number, iDBLanguage, 619 instanceOfName);
    ...
}
```

Voir aussi

[Accéder aux surveillances et aux propriétés du FB ProDiag \(Page 285\)](#)

7.18.6.9 Accéder aux surveillances et aux propriétés du FB ProDiag

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Accéder aux surveillances du FB utilisateur (User-FB)

L'utilisateur d'Openness peut accéder aux surveillances du FB avec l'extrait de code suivant. Chaque FB a une liste de surveillance incluant des API Classic et Plus.

Code de programme : accéder aux surveillances du FB ProDiag

```
...
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");
...
```

Accéder aux attributs du bloc FB

L'utilisateur d'Openness peut définir AssignedProDiagFB dans InstanceDB via l'attribut AssignedProDiagFB (voir Exporter des blocs (Page 448)). L'utilisateur peut accéder aux attributs avec la méthode GetAttribute(), GetAttributes() and SetAttribute(). L'utilisateur ne peut pas utiliser la méthode SetAttributes() pour définir des attributs pour plus d'un attribut. Openness déclenche une exception pour l'utilisation de la méthode SetAttributes() .

Si l'attribut n'est pas pris en charge (dans le bloc indiqué), une exception utilisateur récupérable est déclenchée. Si aucun bloc ProDiag affecté n'est défini, GetAttribute() fournit une chaîne de caractères vide en retour.

Code de programme : appeler et définir le FB ProDiag affecté et l'IDB

```
...
PlcBlockGroup blockFolder = YourUtilities.GetFolder();
PlcSoftware instanceDB = blockFolder.Blocks.Find("IDB");
PlcSoftware plcProdiag = blockFolder.Blocks.Find("block_Prodiag");
instanceDB.SetAttribute("AssignedProDiagFB", plcProdiag.name);
...
```

Voir aussi

Créer un FB ProDiag (Page 284)

7.18.6.10 Supprimer un groupe pour blocs

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Code du programme

Pour supprimer un groupe pour blocs, modifiez le code de programme suivant :

```
// 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();
    }
}
```

Voir aussi

Importation de données de configuration (Page 377)

7.18.6.11 Interroger un groupe système pour blocs système

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme :

Pour déterminer le groupe créé par le système pour les blocs système, modifiez le code de programme suivant :

```
PlcSoftware plcSoftware = ...
foreach (PlcSystemBlockGroup systemGroup in plcSoftware.BlockGroup.SystemBlockGroups)
{
    foreach (PlcSystemBlockGroup group in systemGroup.Groups)
    {
        PlcBlockComposition pbComposition = group.Blocks;
        foreach (PlcBlock block in pbComposition)
        {
            //Ajoutez votre code ici
        }
    }
}
```

7.18.6.12 Enumérer les sous-groupes système

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme : Enumérer tous les sous-groupes système

Pour énumérer les sous-groupes système de tous les blocs système, modifiez le code de programme suivant :

```
//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);
    }
}
```

Code du programme : Accéder à un sous-groupe déterminé

Pour accéder à un sous-groupe déterminé, modifiez le code de programme suivant :

```
private static void AccessSbGroup(PlcSystemBlockGroup systemBlockGroup)
{
    PlcSystemBlockGroup group1 = systemBlockGroup.Groups.Find("User group XYZ");
    PlcSystemBlockGroup group2 = group1.Groups.Find("User group ZYX");
}
```

Voir aussi

Ajouter un fichier externe (Page 288)

7.18.6.13 Ajouter un fichier externe**Conditions requises**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet par le biais d'une application TIA Portal Openness :
Voir Ouvrir un projet (Page 99)

Utilisation

Vous pouvez ajouter un fichier externe à un API. Ce fichier externe est enregistré sous le chemin défini dans le système de fichiers.

Les formats suivants sont pris en charge :

- LIST
- SCL
- DB
- UDT

Remarque

L'accès à des groupes dans le dossier "Fichiers sources externes" n'est pas pris en charge.

Une exception se déclenche si vous indiquez une autre extension de fichier que *.AWL, *.SCL, *.DB ou *UDT.

Code de programme

Pour créer un fichier externe dans le dossier "Fichiers sources externes" à partir d'un bloc, modifiez le code de programme suivant :

```
private static void CreateBlockFromFile(PlcSoftware plcSoftware)
// Creates a block from a AWL, SCL, DB or UDT file
{
    PlcExternalSource externalSource =
plcSoftware.ExternalSourceGroup.ExternalSources.CreateFromFile("SomeBlockNameHere", "SomePa
thHere");
}
```

7.18.6.14 Générer une source à partir d'un bloc

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Utilisation

L'interface TIA Portal Openness API prend en charge la génération de sources en UTF-8 à partir de blocs LIST ou SCL, de blocs de données et de types de données API (types de données utilisateur). Pour générer un fichier source d'un bloc, appelez la méthode `GenerateSource` sur l'instance `PlcExternalSourceSystemGroup`.

L'étendue du fichier source généré dépend de l'option de génération de cette fonction :

- `GenerateOptions.None`
Générer la source uniquement à partir des blocs mis à disposition.
- `GenerateOptions.WithDependencies`
Générer la source y compris tous les objets dépendants.

L'interface `Siemens.Engineering.SW.ExternalSources.IGenerateSource` indique qu'une source peut être générée.

Pour les blocs, seuls les langages de programmation LIST et SCL sont pris en charge. Des exceptions sont lancées dans les cas suivants :

- Le langage de programmation n'est pas LIST ou SCL
- Un fichier du même nom existe déjà au lieu de sauvegarde cible.

Seule l'extension de fichier `"*.udt"` est prise en charge pour les types de données utilisateur. Des exceptions sont lancées dans les cas suivants :

- L'extension de fichier pour blocs de données n'est pas `"*.db"`
- L'extension de fichier pour blocs LIST n'est pas `"*.awl"`
- L'extension de fichier pour blocs SCL n'est pas `"*.scl"`

Code de programme

Pour générer les fichiers sources à partir de blocs et de types, modifiez le code de programme suivant :

```
//method declaration
...
PlcExternalSourceSystemGroup.GenerateSource

(IEnumerable<Siemens.Engineering.SW.ExternalSources.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.
...
```

Voir aussi

Importation de données de configuration (Page 377)

7.18.6.15 Générer les blocs à partir de la source

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Utilisation

Vous pouvez générer des blocs à partir de tous les fichiers externes du groupe "Fichiers sources externes". Seuls les fichiers externes au format ASCII sont pris en charge.

Remarque

L'accès à des groupes dans le dossier "Fichiers sources externes" n'est pas pris en charge.

Les blocs existants sont écrasés.

Si une erreur apparaît lors de l'appel, une Exception est déclenchée. Les 256 premiers caractères de tout message d'erreur sont compris dans le message de l'Exception. Le projet est remis à l'état de traitement où il se trouvait avant l'exécution de la méthode

`GenerateBlocksFromSource`.

Code du programme

Pour générer les blocs à partir de tous les fichiers externes du groupe "Fichiers sources externes", modifiez le code de programme suivant.

```
// Creates a block from an external source file
PlcSoftware plcSoftware = ...;
foreach (PlcExternalSource plcExternalSource in
plcSoftware.ExternalSourceGroup.ExternalSources)
{
    plcExternalSource.GenerateBlocksFromSource();
}
```

7.18.6.16 Supprimer un type de données utilisateur

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Code du programme

Pour supprimer un type utilisateur, modifiez le code de programme suivant :

```
private static void DeleteUserDataTypes(PlcSoftware plcSoftware)
{
    PlcTypeSystemGroup typeGroup = plcSoftware.TypeGroup;
    PlcTypeComposition dataTypes = typeGroup.Types;
    PlcType dataType = dataTypes.Find("DataTypeName");
    if (dataType != null)
    {
        dataType.Delete();
    }
}
```

Voir aussi

Importation de données de configuration (Page 377)

7.18.6.17 Supprimer un fichier externe

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établissement d'une connexion au portail TIA (Page 74)
- Vous avez ouvert un projet par le biais d'une application TIA Portal Openness :
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Code du programme

Pour supprimer un fichier externe dans le groupe "Fichiers sources externes", modifiez le code de programme suivant :

Remarque

L'accès à des groupes dans "Fichiers sources externes" n'est pas pris en charge.

```
// Deletes an external source file
private static void DeleteExternalSource(PlcSoftware plcSoftware)
{
    PlcExternalSource externalSource =
    plcSoftware.ExternalSourceGroup.ExternalSources.Find("myExternalsource");
    externalSource.Delete();
}
```

7.18.6.18 Démarrer un éditeur de bloc

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'instance du portail TIA est ouverte avec interface utilisateur.

Code du programme

Pour démarrer l'éditeur correspondant à une référence d'objet du type `PlcBlock` dans l'instance de TIA Portal, modifiez le code de programme suivant :

```
//Opens a block in a block editor
private static void StartBlockEditor(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    plcBlock.ShowInEditor();
}
```

Pour ouvrir l'éditeur correspondant à une référence d'objet du type `PlcType` dans l'instance de TIA Portal, modifiez le code de programme suivant :

```
//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();
}
```

Voir aussi

Importation de données de configuration (Page 377)

7.18.7 Objets technologiques

7.18.7.1 Vue d'ensemble des objets technologiques

TIA Portal Openness prend en charge une sélection de fonctions d'objets technologiques pour des tâches définies, que vous pouvez appeler à l'aide de l'API Public à l'extérieur de TIA Portal.

Vous recevez les composants de code qui doivent être adaptés pour chaque tâche.

Fonctions

Les fonctions suivantes sont disponibles pour les objets technologiques :

- Interroger la composition des objets technologiques (Page 298)
- Créer un objet technologique (Page 298)
- Supprimer des objets technologiques (Page 299)
- Compiler un objet technologique (Page 300)
- Enumérer des objets technologiques (Page 301)
- Rechercher un objet technologique (Page 302)
- Enumérer les paramètres d'un objet technologique (Page 302)
- Rechercher les paramètres d'un objet technologique (Page 303)
- Lire les paramètres d'un objet technologique (Page 304)
- Ecrire les paramètres d'un objet technologique (Page 305)

7.18.7.2 Vue d'ensemble des objets technologiques et des versions

Objets technologiques

Le tableau suivant indique les objets technologiques disponibles dans l'API Public.

CPU	FW	Objet technologique	Version de l'objet technologique
S7-1200	≥ V4.2	TO_PositioningAxis	V6.0
		TO_CommandTable	
		PID_Compact	V2.3
		PID_3Step	
		PID_Temp	V1.1

CPU	FW	Objet technologique	Version de l'objet technologique
S7-1500	< V2.0	High_Speed_Counter	V3.0
		SSI_Absolute_Encoder	V2.0
	≥ V2.0	TO_SpeedAxis	≥ V3.0
		TO_PositioningAxis	
		TO_ExternalEncoder	
		TO_SynchronousAxis	
		TO_OutputCam	
		TO_CamTrack	
		TO_MeasuringInput	
		TO_Cam (S7-1500T) ¹⁾	
		TO_Kinematics (S7-1500T)	V4.0
		High_Speed_Counter	≥ V3.0
		SSI_Absolute_Encoder	≥ V2.0
		PID_Compact	≥ V2.3
		PID_3Step	V2.3
		PID_Temp	V1.1
		CONT_C	V1.1
	CONT_S		
	TCONT_CP		
TCONT_S			
S7-300/400	Chaque	CONT_C	V1.1
		CONT_S	
		TCONT_CP	
		TCONT_S	
		TUN_EC ²⁾	
		TUN_ES ²⁾	
		PID_CP ²⁾	V2.0
		PID_ES ²⁾	
		AXIS_REF	V2.0

1) L'objet technologique ne prend pas en charge les fonctions Openness suivantes : écriture de paramètres.

2) L'objet technologique ne prend pas en charge les fonctions Openness suivantes : énumération de paramètres, recherche de paramètres, lecture de paramètres, écriture de paramètres.

Remarque

S7-1500 Motion Control

Les objets technologiques TO_OutputCam, TO_CamTrack et TO_MeasuringInput de la S7-1500 sont traités séparément.

Pour plus d'informations, reportez-vous au paragraphe "S7-1500 Motion Control (Page 314)".

7.18.7.3 Vue d'ensemble des types de données

Les types de données des paramètres des objets technologiques dans TIA Portal sont affectés à des types de données C#-dans l'API Public.

Types de données

Le tableau suivant indique l'affectation des types de données :

Format	Type de données dans TIA Portail	Type de données dans C#
Nombres binaires	Bool	bool
	BBool	bool
	Byte	byte
	Word	ushort
	DWord	uint
	LWord	ulong
Nombres entiers	SInt	sbyte
	Int	short
	Dint	int
	LInt	long
	USInt	byte
	UInt	ushort
	UDint	uint
	ULInt	ulong
Nombres à virgule flottante	Real	float
	LReal	double
	Time	double
Chaînes de caractères	Char	char
	WChar	char
	String	string
	WString	string
Types de données matériel	HW_*	ushort
	Block_*	ushort

* réservation pour les extensions du type d'appareil dans le projet TIA Portal.

7.18.7.4 Interroger la composition des objets technologiques

Conditions requises

- L'application Openness est connectée au portail TIA.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT

Code du programme

Pour obtenir tous les objets technologiques d'un API, modifiez le code du programme suivant :

```
// Retrieves all technology objects of a PLC
private static void GetTechnologicalObjectsOfPLC(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBGGroup technologicalObjectGroup =
plcSoftware.TechnologicalObjectGroup;
    TechnologicalInstanceDBComposition technologicalObjects =
technologicalObjectGroup.TechnologicalObjects;
}
```

7.18.7.5 Créer un objet technologique

Conditions requises

- L'application Openness est liée avec le TIA Portal .
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT

Utilisation

Seuls les objets technologiques qui sont listés au paragraphe Vue d'ensemble des objets technologiques et des versions (Page 295) peuvent être créés. Pour les objets technologiques non pris en charge ou les paramètres invalides, une exception est signalée.

Remarque**S7-1500 Motion Control**

Les objets technologiques TO_OutputCam, TO_CamTrack et TO_MeasuringInput de la S7-1500 sont traités séparément.

Pour plus d'informations, reportez-vous au paragraphe "S7-1500 Motion Control (Page 314)".

Code du programme

Pour ajouter un objet technologique depuis un API précédent, modifiez le code du programme suivant :

```
// 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);
}
```

Les valeurs et les combinaisons de nom, type et version des objets technologiques peuvent être trouvées au paragraphe Vue d'ensemble des objets technologiques et des versions (Page 295).

7.18.7.6 Supprimer des objets technologiques**Conditions requises**

- L'application Openness est connectée au portail TIA.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT
- L'objet technologique est présent.
Voir Rechercher un objet technologique (Page 302)

Code du programme

Pour supprimer un objet technologique, modifiez le code du programme suivant :

```
// Delete a technology object from DB composition and from PLC
private static void DeleteTechnologicalObject(TechnologicalInstanceDB technologicalObject)
{
    technologicalObject.Delete();
}
```

7.18.7.7 Compiler un objet technologique

Conditions requises

- L'application Openness est liée à TIA Portal.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT
- L'objet technologique est présent.
Voir Créer un objet technologique (Page 298)

Code du programme Compilation d'un objet technologique

Pour compiler un objet technologique, modifiez le code du programme suivant :

```
// Compile a single technology object
private static void CompileSingleTechnologicalObject(TechnologicalInstanceDB
technologicalObject)
{
    ICompilable singleCompile = technologicalObject.GetService<ICompilable>();
    CompilerResult compileResult = singleCompile.Compile();
}
```

Code du programme Compilation d'un groupe d'objets technologiques

Pour compiler un groupe d'objets technologiques, modifiez le code du programme suivant :

```
// Compile technology object group
private static void CompileTechnologicalObjectGroup(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBGroup technologicalObjectGroup =
plcSoftware.TechnologicalObjectGroup;
    ICompilable groupCompile = technologicalObjectGroup.GetService<ICompilable>();
    CompilerResult compileResult = groupCompile.Compile();
}
```

Résultats de la compilation

Les résultats de la compilation d'objets technologiques sont enregistrés de façon récurrente.

Vous trouverez un exemple du traitement récurrent des résultats de compilation au paragraphe "AUTOHOTSPOT".

7.18.7.8 Enumérer des objets technologiques

Conditions requises

- L'application Openness est connectée au portail TIA.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT

Code du programme

Pour énumérer des objets technologiques, modifiez le code du programme suivant :

```
// Enumerate all technology objects
private static void EnumerateTechnologicalObjects(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBComposition technologicalObjects =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects;
    foreach (TechnologicalInstanceDB technologicalObject in technologicalObjects)
    {
        // Do something ...
    }
}
```

7.18.7.9 Rechercher un objet technologique

Conditions requises

- L'application Openness est connectée au portail TIA.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT

Code du programme

Pour rechercher un objet technologique spécifique, modifiez le code du programme suivant.

```
// 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);
}
```

7.18.7.10 Enumérer les paramètres d'un objet technologique

Conditions requises

- L'application Openness est connectée au portail TIA.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT
- Un objet technologique est présent.
Voir Créer un objet technologique (Page 298) ou Rechercher les paramètres d'un objet technologique (Page 303)
- L'objet technologique (Page 295) prend en charge cette fonction.

Code du programme

Pour énumérer les paramètres d'un objet technologique, modifiez le code du programme suivant :

```
// 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 ...
    }
}
```

Voir aussi

Rechercher un objet technologique (Page 302)

7.18.7.11 Rechercher les paramètres d'un objet technologique

Conditions requises

- L'application Openness est connectée au portail TIA.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT
- Un objet technologique est présent.
Voir Créer un objet technologique (Page 298)
- L'objet technologique (Page 295) prend en charge cette fonction.

Code du programme

Pour rechercher les paramètres d'un objet technologique, modifiez le code du programme suivant :

```
// 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);
}
```

Paramètres de différents objets technologiques

Paramètres de SIMATIC S7-1200 Motion Control (Page 306)

Paramètres de SIMATIC S7-1500 Motion Control (Page 314)

Paramètres de la régulation PID (Page 333)

Paramètres de comptage (Page 334)

Paramètres d'Easy Motion Control (Page 334)

Voir aussi

Rechercher un objet technologique (Page 302)

7.18.7.12 Lire les paramètres d'un objet technologique

Conditions requises

- L'application Openness est connectée au portail TIA.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT
- Un objet technologique est présent.
Voir Créer un objet technologique (Page 298)
- L'objet technologique (Page 295) prend en charge cette fonction.

Code du programme

Pour lire les paramètres d'un objet technologique défini, modifiez le code du programme suivant :

```
// 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;
}
```

Voir aussi

Rechercher un objet technologique (Page 302)

7.18.7.13 Ecrire les paramètre d'un objet technologique

Conditions requises

- L'application Openness est connectée au portail TIA.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Un API est déterminé dans le projet.
Voir AUTOHOTSPOT
- Un objet technologique est présent.
Voir Créer un objet technologique (Page 298)
- L'objet technologique (Page 295) prend en charge cette fonction.

Exception

Une EngineeringException est signalée lorsque :

- vous définissez une nouvelle valeur pour un paramètre qui ne possède pas d'accès en écriture.
- une nouvelle valeur pour un paramètre est d'un type non pris en charge.

Code du programme

Pour écrire un paramètre d'un objet technologique spécifique, modifiez le code du programme suivant :

```
// 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;
}
```

Paramètres des différents objets technologiques

Paramètres de SIMATIC S7-1200 Motion Control (Page 306)

Paramètres de SIMATIC S7-1500 Motion Control (Page 314)

Paramètres de la régulation PID (Page 333)

Paramètres de comptage (Page 334)

Paramètres d'Easy Motion Control (Page 334)

Voir aussi

Rechercher un objet technologique (Page 302)

7.18.7.14 S7-1200 Motion Control

Modifier la version d'Openness Engineering Library

Tant que vous utilisez "Openness\PublicAPI\V14 SP1\Siemens.Engineering.dll" avec TIA Portal V15, votre application Openness actuelle fonctionne normalement.

Lorsque vous passez à "Openness\PublicAPI\V15\Siemens.Engineering.dll" avec TIA Portal V15, vous devez adapter tous les accès aux variables ARRAY pour S7-1200 Motion Control.

Les variables de type ARRAY pour TO_PositioningAxis concernées sont énumérées dans les tableaux suivants :

Accès à Openness < V15	Accès à Openness ≥ V15
_Sensor.Sensor[1].<toutes les variables>	_Sensor[1].<toutes les variables>
ControlPanel.Input.Command.Command[1].<toutes les variables>	ControlPanel.Input.Command[1].<toutes les variables>
ControlPanel.Output.Command.Command[1].<toutes les variables>	ControlPanel.Output.Command[1].<toutes les variables>
Internal.Internal[n].<toutes les variables>	Internal[n].<toutes les variables>
Sensor.Sensor[1].<toutes les variables>	Sensor[1].<toutes les variables>
StatusSensor.StatusSensor[1].<toutes les variables>	StatusSensor[1].<toutes les variables>

Les variables de type ARRAY pour TO_CommandTable concernées sont énumérées dans les tableaux suivants :

Accès à Openness < V15	Accès à Openness ≥ V15
Command.Command[n].<toutes les variables>	Command[n].<toutes les variables>

Connecter des PROFIdrives avec l'adresse matérielle

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1200 est créée dans le projet.
- Un PROFIdrive est disponible dans le projet et connecté à la CPU S7-1200.
- L'objet technologique est présent.
Voir Créer un objet technologique (Page 298).

Code du programme

Modifiez le code de programme suivant pour connecter un PROFIdrive à l'objet technologique "TO_PositioningAxis" à l'aide d'une adresse matérielle.

```
//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 adress 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;
}
```

Connecter un codeur pour PROFIdrives à l'adresse matérielle

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1200 est créée dans le projet.
- Un PROFIdrive est disponible dans le projet et connecté à la CPU S7-1200.
- L'objet technologique est présent.
Voir Créer un objet technologique (Page 298).

Code du programme

Modifiez le code de programme suivant pour connecter un codeur à l'adresse matérielle avec l'objet technologique "TO_PositioningAxis" :

```
//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 adress 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;
}
```

Connecter des entraînements analogiques à l'adresse matérielle

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1200 est créée dans le projet.
- Un entraînement analogique est disponible dans le projet et connecté à la CPU S7-1200.
- L'objet technologique est présent.
Voir Créer un objet technologique (Page 298).

Code du programme

Modifiez le code de programme suivant pour connecter un entraînement analogique à l'adresse matérielle avec l'objet technologique "TO_PositioningAxis" :

```
//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 adress of drive
    technologicalObject.Parameters.Find("_Actor.Interface.Analog").Value = "%QW64";
}
```

Connecter les codeurs pour les entraînements analogiques à l'adresse matérielle

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1200 est créée dans le projet.
- Un entraînement analogique est disponible dans le projet et connecté à la CPU S7-1200.
- L'objet technologique est présent.
Voir Créer un objet technologique (Page 298).

Code du programme

Modifiez le code de programme suivant pour connecter un codeur à l'adresse matérielle avec l'objet technologique "TO_PositioningAxis" :

```
//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;
}
}
```

Connecter des entraînements à un bloc de données

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1200 est créée dans le projet.

- Un bloc de données est disponible dans le projet et réglé sur "Non optimisé".
Avec un type d'axe PROFIdrive, le bloc de données contient une variable de ce type, par exemple PD_TEL3.
Avec un entraînement analogique, le bloc de données contient une variable du type de données Mot.
- L'objet technologique est présent.
Voir Créer un objet technologique (Page 298).

Code du programme

Modifiez le code de programme suivant pour connecter un PROFIdrive à un bloc de données avec l'objet technologique "TO_PositioningAxis".

```
//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";
}
```

Code du programme

Modifiez le code de programme suivant pour connecter un entraînement analogique à un bloc de données avec l'objet technologique "TO_PositioningAxis" :

```
//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";
}
```


Connecter un codeur à un bloc de données

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1200 est créée dans le projet.
- Un bloc de données est disponible dans le projet et réglé sur "Non optimisé".
Avec PROFIdrive, le bloc de données contient une variable de ce type, par exemple PD_TEL3.
- L'objet technologique est présent.
Voir Créer un objet technologique (Page 298).

Code du programme

Modifiez le code de programme suivant pour connecter un codeur à un bloc de données :

```
//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";
}
```

Paramètres pour TO_PositioningAxis et TO_CommandTable

Pour une liste contenant toutes les variables disponibles, voir la description fonctionnelle SIMATIC STEP 7 S7-1200 Motion Control sur Internet (<https://support.industry.siemens.com/cs/ww/fr/view/109741731>)

Remarque

Dans TIA Portal, la colonne "Nom dans Openness" se trouve dans la vue des paramètres de la configuration de l'objet technologique.

7.18.7.15 S7-1500 Motion Control

Créer et trouver TO_OutputCam, TO_CamTrack et TO_MeasuringInput

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1500 est créée dans le projet.
- Un objet technologique du type TO_PositioningAxis, TO_SynchronousAxis ou TO_ExternalEncoder est créé dans le projet.

Application

Les objets technologiques Came, Piste de came et Palpeur de mesure sont connectés à l'un des objets technologiques Axe de positionnement, Axe synchrone ou Codeur externe. Pour accéder aux objets technologiques Came, Piste de came ou Palpeur de mesure, utilisez le service OutputCamMeasuringInputContainer.

Code du programme : Créer et trouver les objets technologiques Came, Piste de came et Palpeur de mesure

Modifiez le code de programme suivant pour créer ou trouver l'objet technologique Came, Piste de came ou Palpeur de mesure.

```

/*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));
}

```

Paramètres de S7-1500 Motion Control

La plupart des paramètres des objets technologiques S7-1500 Motion Control sont représentés directement sur des variables de bloc de données mais il existe également quelques autres paramètres qui ne le sont pas. Dans Openness, les paramètres représentés directement ont le même ordre que dans la "navigation des données" dans la vue des paramètres de l'objet technologique. Les paramètres supplémentaires suivent les paramètres représentés directement dans l'ordre du tableau.

Paramètres directement représentés sur les variables de bloc de données de l'objet technologique :

Vous avez accès à toutes les variables de bloc de données de l'objet technologique généralement décrites sauf :

- Variables protégées en écriture
- Variables du type de données VREF
- Variables avec la structure "InternalToTrace"
- Variables avec la structure "ControlPanel"

Vous pouvez trouver dans les annexes suivantes des informations supplémentaires sur les paramètres représentés directement :

- Description fonctionnelle SIMATIC S7-1500 Motion Control : <https://support.industry.siemens.com/cs/ww/fr/view/109739589> (<https://support.industry.siemens.com/cs/ww/fr/view/109739589>)
- Description fonctionnelle SIMATIC S7-1500T Motion Control: <https://support.industry.siemens.com/cs/ww/fr/view/109481326> (<https://support.industry.siemens.com/cs/ww/fr/view/109481326>)

Certains paramètres technologiques représentant des variables de bloc de données doivent être accessibles en écriture dans le PublicAPI. Les valeurs autorisées sont les mêmes que pour les variables de bloc de données sous-jacentes. Les paramètres concernés sont énumérés dans les tableaux suivants :

Nom dans Openness	Type de données	TO_SpeedAxis	TO_PositioningAxis	TO_SynchronousAxis	TO_ExternalEncoder
Actor.Type	int	X	X	X	
Actor.Interface.EnableDriveOutput	bool	X	X	X	
Actor.Interface.DriveReadyInput	bool	X	X	X	
VirtualAxis.Mode	uint	X	X	X	
Sensor[n].Existent ¹⁾	bool		X	X	
Sensor[n].Interface.Number ¹⁾	uint		X	X	
Sensor[n].Type ¹⁾	int		X	X	
Sensor.Interface.Number	uint				X
Sensor.Type	int				X

Nom dans Openness	Type de données	TO_OutputCam	TO_MeasuringInput	TO_Kinematic ²⁾
Interface.LogicOperation	int	X		
Parameter.MeasuringInputType	int		X	
Kinematics.TypeOfKinematics	int			X
MotionQueue.MaxNumberOfCommands	int			X

1) CPU S7-1500 : n=1 ; CPU S7-1500T : 1≤n≤4

2) CPU S7-1500T

Paramètres qui ne sont pas directement représentés sur les variables de bloc de données de l'objet technologique :

Les paramètres supplémentaires suivants qui ne sont pas directement représentés sur les variables de bloc de données sont disponibles pour les objets technologiques S7-1500 Motion Control :

Nom dans Openness	Nom dans la vue de fonction	Valeur possible	Type de données dans Openness	TO_SpeedAxis	TO_PositioningAxis TO_SynchronousAxis	TO_ExternalEncoder
_Properties.MotionType	Type d'axe ou "unité technique de la position"	0 : Linéaire 1 : Rotatif	int		X	X
_Units.LengthUnit	Unités de position	Voir Variable Units.LengthUnit ²⁾	uint		X	X
_Units.VelocityUnit	Unités de vitesse	Voir Variable Units.VelocityUnit ²⁾	uint	X	X	X
_Units.TorqueUnit	Unités de couple	Voir Variable Units.TorqueUnit ²⁾	uint	X	X	
_Units.ForceUnit	Unités de force	Voir Variable Units.ForceUnit ²⁾	uint		X	
_Actor.Interface.Telegram	Télégramme d'entraînement	Numéro de télégramme ³⁾	uint	X	X	
_Actor.Interface.EnableDriveOutputAddress	Adresse pour la sortie Validation d'entraînement	Objet PublicAPI	SW.Tags.PlcTag	X	X	
_Actor.Interface.DriveReadyInputAddress	Adresse pour l'entrée Validation d'entraînement	Objet PublicAPI	SW.Tags.PlcTag	X	X	
_Sensor[n].Interface.Telegram ⁴⁾	Télégramme de codeur	Numéro de télégramme ³⁾	uint		X	
_Sensor[n].ActiveHoming.DigitalInputAddress ⁴⁾	Entrée TOR	Objet PublicAPI	SW.Tags.PlcTag		X	
_Sensor[n].PassiveHoming.DigitalInputAddress ⁴⁾	Entrée TOR	Objet PublicAPI	SW.Tags.PlcTag		X	
_PositionLimits_HW.MinSwitchAddress	Adresse du fin de course matériel négatif	Objet PublicAPI	SW.Tags.PlcTag		X	

7.18 Fonctions sur les données d'un appareil API

Nom dans Openness	Nom dans la vue de fonction	Valeur possible	Type de données dans Openness	TO_SpeedAxis	TO_PositioningAxis TO_SynchronousAxis	TO_ExternalEncoder
_PositionLimits_HW.MaxSwitchAddress	Adresse du fin de course matériel positif	Objet PublicAPI	SW.Tags.PlcTag		X	
_Sensor.Interface.Telemgram	Télégramme de codeur	Numéro de télégramme ³⁾	uint			X
_Sensor.PassiveHoming.DigitalInputAddress	Entrée TOR	Objet PublicAPI	SW.Tags.PlcTag			X

Le paramètre supplémentaire suivant est disponible pour les objets technologiques Came, Piste de came et Palpeur de mesure :

Nom dans Openness	Nom dans la vue de fonction	Valeur possible	Type de données
_AssociatedObject	Axe ou codeur externe affecté	Objet PublicAPI	SW.TechnologicalObjects.TechnologicalInstanceDB

Pour l'objet technologique Cinématique, les paramètres supplémentaires suivants sont disponibles (S7-1500T) :

Nom dans Openness	Nom dans la vue de fonction	Valeur possible	Type de données
_KinematicsAxis[1...4]	Axes 1 - 3, axe d'orientation	Axe pouvant être connecté aux objets TO_Kinematics	SW.TechnologicalObjects.TechnologicalInstanceDB
_Units.LengthUnit	Unité de mesure > position	Voir Variable Units.LengthUnit ²⁾	uint
_Units.LengthVelocityUnit	Unité de mesure > vitesse	Voir Variable Units.LengthVelocityUnit ²⁾	uint
_Units.AngleUnit	Unité de mesure > angle	Voir Variable Units.AngleUnit ²⁾	uint
_Units.AngleVelocityUnit	Unité de mesure > vitesse angulaire	Voir Variable Units.AngleVelocityUnit ²⁾	uint

2) Les valeurs possibles sont décrites dans le chapitre Variable Units (TO) de la description fonctionnelle S7-1500 Motion Control

3) Les valeurs possibles sont décrites dans le chapitre Télégrammes PROFIdrive de la description fonctionnelle S7-1500 Motion Control

4) CPU S7-1500 : n=1 ; CPU S7-1500T : 1≤n≤4

Code du programme : Variables de bloc de données directement représentées

Pour accéder aux paramètres représentés directement, modifiez le code de programme suivant :

```
//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;
}
```

Code du programme : Autres paramètres

Pour accéder aux autres paramètres, modifiez le code de programme suivant :

```
//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;
}
```

Informations supplémentaires

Pour plus d'informations, voir :

- Description fonctionnelle SIMATIC S7-1500 Motion Control :
<https://support.industry.siemens.com/cs/ww/fr/view/109739589> (<https://support.industry.siemens.com/cs/ww/fr/view/109739589>)
- Description fonctionnelle SIMATIC S7-1500T Motion Control :
<https://support.industry.siemens.com/cs/ww/fr/view/109481326> (<https://support.industry.siemens.com/cs/ww/fr/view/109481326>)

Connecter des entraînements

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.

- Une CPU S7-1500 est créée dans le projet.
- Un objet technologique du type TO_SpeedAxis, TO_PositioningAxis ou TO_SynchronousAxis est créé dans le projet.
- Un entraînement est créé dans le projet.

Application

Pour connecter un axe à un entraînement, il est nécessaire d'indiquer plusieurs valeurs ensemble au cours d'un seul appel. Le type du Public API AxisEncoderHardwareConnectionInterface offre les méthodes suivantes pour connecter ou couper les interfaces d'actionneur ou de capteur :

Méthode	Description
void Connect(HW.Deviceltem moduleInOut)	Connecte aux adresses d'entrée et de sortie d'un module
void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut)	Connecte aux adresses d'entrée et de sortie de modules coupés
void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut, ConnectOption connectOption)	Connecte aux adresses d'entrée et de sortie de modules coupés en indiquant une ConnectOption supplémentaire
void Connect(HW.Channel channel)	Connecte à une voie
void Connect(int addressIn, int addressOut, ConnectOption connectOption)	Connecte des adresses binaires directement
void Connect(string pathToDBMember)	Connecte à une variable de bloc de données
void Connect(SW.Tags.PlcTag outputTag)	Connecte à une variable CPU
void Disconnect()	Coupe une connexion existante

Remarque

Connexion automatique

Notez que s'applique ici le même comportement que dans l'interface utilisateur. Les parties sont toujours connectées automatiquement lorsque l'interface d'actionneur est connectée via l'une des méthodes de connexion suivantes et que le télégramme contient une partie capteur ou télégramme 750.

Vous pouvez utiliser les attributs en lecture seule suivants pour déterminer comment l'objet technologique est connecté. Les valeurs de connexion correspondantes sont réglées uniquement si une connexion spécifique de ce type existe.

Attribut	Type de données	Description
IsConnected	bool	TRUE : L'interface est connectée FALSE : L'interface n'est pas connectée
InputOutputModule	HW.Deviceltem	Module connecté qui contient des adresses d'entrée et de sortie
InputModule	HW.Deviceltem	Module connecté qui contient des adresses d'entrée La valeur est également réglée si une connexion à un module contenant des adresses d'entrée et de sortie existe.
OutputModule	HW.Deviceltem	Module connecté qui contient des adresses de sortie La valeur est également réglée si une connexion à un module contenant des adresses d'entrée et de sortie existe.

Attribut	Type de données	Description
InputAddress	int	Adresse d'entrée logique de l'objet connecté ; par exemple : 256.
OutputAddress	int	Adresse de sortie logique de l'objet connecté ; par exemple : 256.
ConnectOption	ConnectOption	Valeur de la ConnectOption paramétrée lors de l'établissement de la connexion : <ul style="list-style-type: none"> • Paramètre par défaut Seuls les modules détectés comme partenaires de liaison valides peuvent être sélectionnés. • AllowAllModules Correspond à la sélection de "Afficher tous les modules" dans l'interface utilisateur.
Channel	HW.Channel	Voie connectée
PathToDBMember	string	Variable de bloc de données de l'objet technologique connectée
OutputTag	SW.Tags.PlcTag	Variable CPU connectée (connexion analogique)
SensorIndexInActor-Telegram	int	Partie capteur connectée dans le télégramme d'activation Cet attribut n'est pertinent que pour les interfaces de capteur. 0 : Le codeur n'est pas connecté 1 : Le codeur est connecté à la première interface de capteur dans le télégramme 2 : Le codeur est connecté à la seconde interface de capteur dans le télégramme La valeur de l'interface d'actionneur est toujours 0.

Remarque**Accès à l'interface de capteur**

Pour accéder à l'interface de capteur, vous pouvez utiliser `SensorInterface[m]` avec $0 \leq m \leq 3$.

Code du programme : void Connect(HW.DeviceItem moduleInOut)

Modifiez le code de programme suivant pour connecter un module mixte contenant des adresses d'entrée et de sortie.

```
//An instance of technology object and device item is already available in the program
before
private static void UseServiceAxisHardwareConnectionProvider(TechnologicalInstanceDB
technologyObject, DeviceItem devItem)
{
    //Retrieve service AxisHardwareConnectionProvider
    AxisHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<AxisHardwareConnectionProvider>();

    //Connect ActorInterface with DeviceItem
    connectionProvider.ActorInterface.Connect(devItem);

    //Connect first SensorInterface with DeviceItem
    connectionProvider.SensorInterface[0].Connect(devItem);

    //Check ConnectionState of ActorInterface
    bool actorInterfaceConnectionState = connectionProvider.ActorInterface.IsConnected;

    //Check ConnectionState of first SensorInterface
    bool sensorInterfaceConnectionState =
    connectionProvider.SensorInterface[0].IsConnected;
}
```

Connecter le télégramme 750

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT
- Un projet est ouvert.
Voir AUTOHOTSPOT
- Une CPU S7-1500 est créée dans le projet.
- Un objet technologique du type TO_SpeedAxis, TO_PositioningAxis ou TO_SynchronousAxis V4.0 est créé dans le projet.
- Un entraînement, qui prend en charge le télégramme 750, est créé dans le projet.

TorqueHardwareConnectionInterface

Application

Lorsque le télégramme 750 est ajouté après la connexion de l'entraînement et de l'axe, le télégramme 750 doit être connecté séparément. EnableTorqueData est automatiquement mis

sur TRUE . Le type du Public API TorqueHardwareConnectionInterface offre les méthodes suivantes pour connecter ou couper le télégramme 750.

Méthode	Description
void Connect(HW.Deviceltem moduleInOut)	Connecte aux adresses d'entrée et de sortie d'un module
void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut)	Connecte aux adresses d'entrée et de sortie de modules coupés
void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut, ConnectOption connectOption)	Connecte aux adresses d'entrée et de sortie de modules coupés en indiquant une ConnectOption supplémentaire
void Connect(int addressIn, int addressOut, ConnectOption connectOption)	Connecte des adresses binaires directement
void Connect(string pathToDBMember)	Connecte à une variable de bloc de données
void Disconnect()	Coupe une connexion existante

TorqueHardwareConnectionInterface peut être appelée via l'attribut TorqueInterface sur le type AxisHardwareConnectionProvider. Lorsque la connexion au télégramme 750 n'est pas prise en charge, l'attribut possède la valeur "zéro".

En cas de connexion de l'entraînement par variables de bloc de données, vous ne pouvez pas connecter le télégramme 750 à l'aide du module. Vous pouvez utiliser les attributs en lecture seule suivants pour déterminer comment l'objet technologique est connecté. Les valeurs de connexion correspondantes sont réglées uniquement si une connexion spécifique de ce type existe.

Attribut	Type de données	Description
IsConnected	bool	TRUE : l'interface est connectée FALSE : l'interface n'est pas connectée
InputOutputModule	HW.Deviceltem	Module connecté qui contient des adresses d'entrée et de sortie
InputModule	HW.Deviceltem	Module connecté qui contient des adresses d'entrée La valeur est également réglée si une connexion à un module contenant des adresses d'entrée et de sortie existe.
OutputModule	HW.Deviceltem	Module connecté qui contient des adresses de sortie La valeur est également réglée si une connexion à un module contenant des adresses d'entrée et de sortie existe.
InputAddress	int	Adresse d'entrée logique de l'objet connecté ; par exemple : 256.
OutputAddress	int	Adresse de sortie logique de l'objet connecté ; par exemple : 256.

Attribut	Type de données	Description
ConnectOption	ConnectOption	Valeur de la ConnectOption paramétrée lors de l'établissement de la connexion : <ul style="list-style-type: none"> • Standard Seuls les modules détectés comme partenaires de connexion valides peuvent être sélectionnés. • AllowAllModules Correspond à la sélection de "Afficher tous les modules" dans l'interface utilisateur.
PathtoDBMember	string	Chemin d'accès connecté au membre DB

Code du programme : void Connect(HW.DeviceItem moduleInOut)

Modifiez le code de programme suivant pour connecter un module mixte contenant des adresses d'entrée et de sortie :

```
//Une instance d'objet technologique et d'élément d'appareil existe déjà dans le programme

private static void ConnectTorqueInterface(TechnologicalInstanceDB
technologyObject, DeviceItem devItem)
{
//Appeler le service AxisHardwareConnectionProvider
AxisHardwareConnectionProvider connectionProvider =
technologyObject.GetService<AxisHardwareConnectionProvider>();
//Connecter TorqueInterface à DeviceItem
connectionProvider.TorqueInterface.Connect(devItem);
```

Connecter un codeur

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1500 est créée dans le projet.
- Un objet technologique du type TO_ExternalEncoder est créé dans le projet.
- Un objet fournissant le télégramme PROFIdrive 81 ou 83 est défini dans le projet.

Application

Pour connecter un objet technologique Codeur externe au matériel du codeur, il est nécessaire d'indiquer plusieurs valeurs ensemble au cours d'un seul appel. Le type du Public API AxisEncoderHardwareConnectionInterface offre les méthodes suivantes pour connecter ou couper l'interface de capteur.

Méthode	Description
void Connect(HW.Deviceltem moduleInOut)	Connecte aux adresses d'entrée et de sortie d'un module
void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut)	Connecte aux adresses d'entrée et de sortie de modules coupés
void Connect(HW.Deviceltem moduleIn, HW.Deviceltem moduleOut, ConnectOption connectOption)	Connecte aux adresses d'entrée et de sortie de modules coupés en indiquant une ConnectOption supplémentaire
void Connect(HW.Channel channel)	Connecte à une voie
void Connect(int addressIn, int addressOut, ConnectOption connectOption)	Connecte des adresses binaires directement
void Connect(string pathToDBMember)	Connecte à une variable de bloc de données
void Connect(SW.Tags.PlcTag outputTag)	Non pertinent pour la connexion de codeurs
void Disconnect()	Coupe une connexion existante

Vous pouvez utiliser les attributs en lecture seule suivants pour déterminer comment l'objet technologique est connecté. Les valeurs de connexion correspondantes sont réglées uniquement si une connexion spécifique de ce type existe.

Attribut	Type de données	Description
IsConnected	bool	TRUE : L'interface est connectée FALSE : L'interface n'est pas connectée
InputOutputModule	HW.Deviceltem	Module connecté qui contient des adresses d'entrée et de sortie
InputModule	HW.Deviceltem	Module connecté qui contient des adresses d'entrée La valeur est également réglée si une connexion à un module contenant des adresses d'entrée et de sortie existe.
OutputModule	HW.Deviceltem	Module connecté qui contient des adresses de sortie La valeur est également réglée si une connexion à un module contenant des adresses d'entrée et de sortie existe.
InputAddress	int	L'adresse d'entrée logique de l'objet connecté est par exemple 256.
OutputAddress	int	L'adresse de sortie logique de l'objet connecté est par exemple 256.
ConnectOption	ConnectOption	Valeur de la ConnectOption paramétrée lors de l'établissement de la connexion : <ul style="list-style-type: none"> • Default Seuls les modules détectés comme partenaires de liaison valides peuvent être sélectionnés. • AllowAllModules Correspond à la sélection de "Afficher tous les modules" dans l'interface utilisateur.
Channel	HW.Channel	Voie connectée
PathToDBMember	string	Variable de bloc de données connectée

Attribut	Type de données	Description
OutputTag	SW.Tags.PlcTag	Non pertinent pour la connexion de codeurs
SensorIndexInActor-Telegram	int	Télégramme de capteur connecté Cet attribut n'est pertinent que pour les interfaces de capteur. 0 : Le codeur n'est pas connecté 1 : Le codeur est connecté à la première interface de capteur dans le télégramme 2 : Le codeur est connecté à la seconde interface de capteur dans le télégramme La valeur de l'interface d'actionneur est toujours 0.

Code du programme : Connecter un codeur

Modifiez le code de programme suivant pour connecter un objet technologique Codeur externe :

```
//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;
}
```

Connecter une came et une piste de came au matériel

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1500 est créée dans le projet.
- Un objet technologique du type TO_OutputCam ou TO_CamTrack est créé dans le projet.
- Un module de sorties TOR est créé dans le projet, par exemple TM Timer DIDQ.

Application

Pour connecter un objet technologique Came ou Piste de came à une sortie TOR, il est nécessaire d'indiquer plusieurs valeurs ensemble au cours d'un seul appel. Le type du Public API OutputCamHardwareConnectionProvider offre les méthodes suivantes pour connecter ou couper les interfaces d'actionneur ou de capteur :

Méthode	Description
void Connect(HW.Channel channel)	Connecte à une voie
void Connect(SW.Tags.PlcTag outputTag)	Connecte à une variable CPU
void Connect(int address)	Connecte des adresses binaires directement
void Disconnect()	Coupe une connexion existante

Vous pouvez utiliser les attributs en lecture seule suivants pour déterminer comment l'objet technologique est connecté :

Attribut	Type de données	Description
IsConnected	bool	TRUE : l'objet technologique est connecté FALSE : l'objet technologique n'est pas connecté
Channel	HW.Channel	Voie connectée
OutputTag	SW.Tags.PlcTag	Variable CPU connectée
OutputAddress	int	L'adresse de sortie logique de l'objet connecté est par exemple 256.

Code du programme : connecter l'objet technologique Came ou Piste de came

Modifiez le code de programme suivant pour connecter un objet technologique Came ou Piste de came :

```
//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;
}
```

Connecter le palpeur de mesure avec le matériel

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1500 est créée dans le projet.
- Un objet technologique du type TO_MeasuringInput est créé dans le projet.
- Un module d'entrées TOR est créé dans l'entraînement ou dans le projet, par exemple TM Timer DIDQ.

Application

Pour connecter un objet technologique Palpeur de mesure à une entrée TOR, il est nécessaire d'indiquer plusieurs valeurs ensemble au cours d'un seul appel. Le type du Public API MeasuringInputHardwareConnectionProvider offre les méthodes suivantes pour connecter ou couper les interfaces d'actionneur ou de capteur :

Méthode	Description
void Connect(HW.Channel channel)	Connecte à une voie
void Connect(HW.DeviceItem moduleIn, int channelIndex)	Connecte à un module et définit un indice de canal de supplémentaire
void Connect(int address)	Connecte des adresses binaires directement
void Disconnect()	Coupe une connexion existante

Vous pouvez utiliser les attributs en lecture seule suivants pour déterminer comment l'objet technologique est connecté :

Attribut	Type de données	Description
IsConnected	bool	TRUE : l'objet technologique est connecté FALSE : l'objet technologique n'est pas connecté
InputModule	HW.DeviceItem	Module connecté qui contient des adresses d'entrée
ChannelIndex	int	Indice du canal connecté concernant le module d'entrées
Channel	HW.Channel	Voie connectée
InputAddress	int	L'adresse d'entrée logique de l'objet connecté est par exemple 256.

Code du programme : connecter l'objet technologique Palpeur de mesure

Modifiez le code de programme suivant pour connecter un objet technologique Palpeur de mesure :

```
//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;
}
```

Connecter un axe synchrone à des valeurs pilote

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Une CPU S7-1500 est créée dans le projet.
- Un objet technologique du type TO_PositioningAxis, TO_SynchronousAxis ou TO_ExternalEncoder en tant qu'axe pilote est créé dans le projet.
- Un objet technologique du type TO_SynchronousAxis en tant qu'axe asservi est créé dans le projet.

Application

Pour connecter un objet technologique Axe synchrone à des valeurs pilote, il est nécessaire d'indiquer plusieurs valeurs ensemble au cours d'un seul appel. Le type du Public API `SynchronousAxisMasterValues` offre les méthodes suivantes pour connecter ou couper des valeurs pilote. Les valeurs pilote peuvent être connectées en tant que couplages par valeur de consigne (CPU S7-1500, CPU S7-1500T) ou en tant que couplages par valeur réelle (CPU S7-1500T). Toutes les méthodes et tous les attributs sont pertinents pour les deux méthodes de couplage.

Méthode	Description
<code>int IndexOf (TechnologicalInstanceDB element)</code>	Renvoie l'indice correspondant d'une valeur pilote
<code>bool Contains (TechnologicalInstanceDB element)</code>	TRUE : le conteneur contient la valeur pilote FALSE : le conteneur ne contient pas la valeur pilote
<code>IEnumerator GetEnumerator <TechnologicalInstanceDB>()</code>	Sert à la prise en charge de chaque itération
<code>void Add (TechnologicalInstanceDB element)</code>	Connecte l'axe asservi à une valeur pilote
<code>bool Remove (TechnologicalInstanceDB element)</code>	Sépare l'axe asservi de la valeur pilote TRUE : la connexion a été coupée avec succès FALSE : impossible de couper la connexion

Vous pouvez utiliser les attributs en lecture seule suivants :

Attribut	Type de données	Description
<code>Count</code>	<code>int</code>	Nombre de valeurs pilote
<code>IsReadOnly</code>	<code>bool</code>	TRUE : le conteneur est protégé en écriture FALSE : le conteneur n'est pas protégé en écriture
<code>Parent</code>	<code>IEngineeringObject</code>	Renvoie les valeurs de niveau supérieur du conteneur. Dans ce cas, la valeur de niveau supérieur est le service <code>SynchronousAxisMasterValues</code> .
<code>this [id] { get; }</code>	<code>TechnologicalInstanceDB</code>	Accès basé sur des indices aux valeurs pilote

Code du programme : Connecter l'axe synchrone à une valeur pilote

Modifiez le code de programme suivant pour connecter un axe synchrone à une valeur pilote :

```
//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);
}
```

Exporter et importer l'objet technologique Cam (S7-1500T)

Conditions

- L'application Openness est connectée à TIA Portal.
Voir AUTOHOTSPOT.
- Un projet est ouvert.
Voir AUTOHOTSPOT.
- Un AP S7-1500 est créé dans le projet.
Voir AUTOHOTSPOT
- L'objet technologique est présent.

Application

Pour exporter ou importer les données d'un objet technologique Cam, vous devez indiquer le format et le séparateur souhaité. Le type du Public API CamDataSupport offre les méthodes suivantes que vous pouvez utiliser pour exporter les données de l'objet technologique Cam :

Méthode	Description
void SaveCamDataBinary (System.IO.FileInfo destinationFile)	Exporte les données au format binaire dans un fichier cible.
void SaveCamDataPointList (System.IO.FileInfo destinationFile, CamDataFormatSeparator separator, int samplePoints)	Exporte les données au format "PointList" dans un fichier cible.
void SaveCamData (System.IO.FileInfo destinationFile, CamDataFormat format, CamDataFormatSeparator separator)	Exporte les données dans un fichier cible. Vous pouvez choisir "MCD", "SCOUT" ou "Pointlist" comme format de données et "Tabulation" ou "Virgule" comme séparateur. Si vous choisissez "PointList", 360 points d'interpolation sont exportés.
void LoadCamData(System.IO.FileInfo sourceFile, CamDataFormatSeparator separator)	Importe les données Cam au format "MCD", "SCOUT" ou "PointList" dans le projet.
void LoadCamDataBinary(System.IO.FileInfo sourceFile)	Importe les données Cam d'un fichier binaire dans le projet.

Vous pouvez utiliser les attributs suivants :

Attribut	Type de données	Description
separator	CamDataFormatSeparator	Valeurs autorisées <ul style="list-style-type: none"> • Tabulation • Virgule
samplePoints	int	Nombre de points d'interpolation à exporter.
format	CamDataFormat	Valeurs autorisées <ul style="list-style-type: none"> • MCD • SCOUT • PointList
destinationFile	System.IO.FileInfo	Nom du fichier cible. Doit être différent de zéro. Les droits d'accès nécessaires doivent être attribués et le support de stockage doit disposer de suffisamment d'espace mémoire. Les fichiers existants sont écrasés.
sourceFile	System.IO.FileInfo	Nom du fichier source. Doit être différent de zéro. Les droits d'accès nécessaires doivent être attribués. Le contenu doit exister dans le format indiqué.

Code du programme : Exporter les données Cam

Modifiez le code de programme suivant pour exporter des données Cam :

```
//Une instance d'objet technologique existe déjà dans le programme
private static void ExportCamData(TechnologicalInstanceDB
technologyObject, System.IO.FileInfo destinationFile)
{
//Appeler le service CamDataSupport
```

```
CamDataSupport camData =  
technologyObject.GetService<CamDataSupport>(); //Enregistrer les  
données Cam au format MCD, utiliser Tabulation comme séparateur  
  
camData.SaveCamData(destinationFile, CamDataFormat.MCD,  
CamDataFormatSeparator.Tab);
```

Code du programme : Importer les données Cam

Modifiez le code de programme suivant pour importer des données Cam :

```
//Une instance d'objet technologique existe déjà dans le programme  
private static void ImportCamData(TechnologicalInstanceDB  
technologyObject, System.IO.FileInfo sourceFile)  
{  
//Appeler le service CamDataSupport  
  
CamDataSupport camData =  
technologyObject.GetService<CamDataSupport>(); //Connecter  
TorqueInterface à DeviceItem  
  
camData.LoadCamData(sourceFile, CamDataFormatSeparator.Tab);  
}
```

7.18.7.16 Régulation PID

Paramètres de PID_Compact, PID_3Step, PID_Temp, CONT_C, CONT_S, TCONT_CP et TCONT_S

Vous trouverez une liste de tous les paramètres disponibles dans l'information produit "Paramètres des objets technologiques dans TIA Portal Openness" sur Internet (<https://support.industry.siemens.com/cs/ww/fr/view/109744932>).

Les propriétés suivantes sont affichées pour chaque paramètre :

- Nom dans la configuration (TIA Portal)
- Nom dans Openness
- Type de données dans Openness
- Accès standard
- Plage de valeurs

Remarque

Dans TIA Portal, vous trouverez dans la vue de paramètre de la configuration d'objet technologique la colonne "Name in Openness".

Pour plus d'informations...

Pour plus d'informations, reportez-vous à la description fonctionnelle SIMATIC S7-1200/S7-1500 PID control sur Internet (<https://support.industry.siemens.com/cs/ww/fr/view/108210036>).

7.18.7.17 Comptage

Paramètres pour High_Speed_Counter et SSI_Absolute_Encoder

Vous trouverez une liste de tous les paramètres disponibles dans l'information produit "Paramètres des objets technologiques dans TIA Portal Openness" sur Internet (<https://support.industry.siemens.com/cs/ww/fr/view/109744932>).

Les propriétés suivantes sont affichées pour chaque paramètre :

- Nom dans la configuration (TIA Portal)
- Nom dans Openness
- Type de données dans Openness
- Accès standard
- Plage de valeurs

Pour plus d'informations...

Pour plus d'informations, reportez-vous à la description fonctionnelle SIMATIC S7-1500, ET 200MP, ET 200SP Comptage, mesure et mesure de déplacement sur Internet (<http://support.automation.siemens.com/WWW/view/fr/59709820>).

7.18.7.18 Easy Motion Control

Paramètres pour AXIS_REF

Vous trouverez une liste de tous les paramètres disponibles dans l'information produit "Paramètres des objets technologiques dans TIA Portal Openness" sur Internet (<https://support.industry.siemens.com/cs/ww/fr/view/109744932>).

Les propriétés suivantes sont affichées pour chaque paramètre :

- Nom dans la configuration (TIA Portal)
- Nom dans Openness
- Type de données dans Openness
- Accès standard
- Plage de valeurs

Remarque

Dans TIA Portal, vous trouverez dans la vue de paramètre de la configuration d'objet technologique la colonne "Name in Openness".

Pour plus d'informations...

Vous trouverez plus d'informations sur Easy Motion Control dans les informations système de STEP 7 (TIA Portal).

7.18.8 Variables et tables de variables**7.18.8.1 Démarrage de l'éditeur de variables API****Conditions requises**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'instance du portail TIA est ouverte avec interface utilisateur.

Code du programme

Pour démarrer l'éditeur correspondant à une référence d'objet du type `PlcTagTable` dans l'instance de TIA Portal, modifiez le code de programme suivant :

```
//Ouvre la table des variables dans l'éditeur "Tags"
private static void OpenTagtableInEditor(PlcSoftware plcSoftware)
{
    PlcTagTable plcTagTable = plcSoftware.TagTableGroup.TagTables.Find("MyTagTable");
    plcTagTable.ShowInEditor();
}
```

Voir aussi

Importation de données de configuration (Page 377)

7.18.8.2 Interroger un groupe système pour variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Une instance PlcSoftware a été appelée par un élément d'appareil API.
Voir Interroger PLC Target et HMI Target (Page 161)

Code du programme

Pour interroger un groupe système pour variables API, modifiez le code de programme suivant :

```
//Retrieves the plc tag table group from a plc
private PlcTagTableSystemGroup GetControllerTagfolder(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    return plcTagTableSystemGroup;
}
```

7.18.8.3 Créer une table des variables API

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Une instance PlcSoftware a été appelée par un élément d'appareil API.
Voir Interroger PLC Target et HMI Target (Page 161)

Code de programme

Pour créer la table des variables API, modifiez le code de programme suivant : Une nouvelle table des variables API est créée avec le nom indiqué dans la combinaison.

```
PlcTagTable myTable = plc.TagTableGroup.TagTables.Create("myTable");
```

Voir aussi

Interroger PLC Target et HMI Target (Page 161)

7.18.8.4 Enumérer les groupes personnalisés pour variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Une instance PlcSoftware a été appelée par un élément d'appareil API.
Voir Interroger PLC Target et HMI Target (Page 161)

Utilisation

Les sous-dossiers compris sont considérés comme récurrents lors de l'énumération.

Code du programme : Enumérer les groupes personnalisés pour variables API

Pour énumérer les groupes personnalisés pour variables API, modifiez le code de programme suivant :

```
//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
    }
}
```

Code du programme : Accéder à un groupe personnalisé

Pour accéder à un groupe personnalisé pour variables API, modifiez le code de programme suivant :

```
//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
}
```

7.18.8.5 Créer les groupes personnalisés pour variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'interface TIA Portal Openness API prend en charge la création d'un groupe personnalisé pour variables API.

Code du programme

Pour créer un groupe personnalisé pour variables API, modifiez le code de programme suivant :

```
//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");
}
```

7.18.8.6 Supprimer les groupes personnalisés pour variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'interface TIA Portal Openness API prend en charge la suppression d'un groupe personnalisé spécifique pour tables de variables API.

Code du programme

Pour supprimer un groupe personnalisé déterminé pour tables de variables API, modifiez le code de programme suivant :

```
private static void DeletePlcTagTableUserGroup(PlcSoftware plcSoftware)
{
    PlcTagTableUserGroup group = plcSoftware.TagTableGroup.Groups.Find("MySubGroupName");
    if (group != null)
    {
        group.Delete();
    }
}
```

7.18.8.7 Enumérer des tables de variables API dans un dossier

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme : Enumérer des tables de variables API

Pour énumérer toutes les tables de variables API contenues dans les groupes système ou groupes personnalisés, modifiez le code de programme suivant :

```
//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
    }
}
```

Code du programme : Accéder à une table de variables API

Pour accéder à la table de variables API de votre choix, modifiez le code de programme suivant :

```
//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
}
```

7.18.8.8 Interroger les informations d'une table de variables API

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les tables de variables API vous permettent d'accéder à des constantes utilisateur, des constantes système et des variables. Le nombre de compositions de variables d'une table des variables correspond au nombre de variables d'une table des variables. La table PLCTagTable comprend les navigateurs, attributs et actions suivants.

Vous avez accès aux attributs suivants dans la table des variables API.

Nom	Type	Type
IsDefault	Bool	Protégé en écriture
ModifiedTime	DateTime	Protégé en écriture
Nom	String	Protégé en écriture

La table des variables API comprend les actions indiquées ci-après.

Nom	Type de valeur en retour	Description
Delete	Vide	Supprime l'instance. Déclenche une exception si IsDefault est vrai.
Exporter	Vide	Exporte SIMATIC ML d'une table des variables API.
ShowInEditor	Vide	Affiche la table des variables dans l'éditeur de table des variables API.

Code de programme

Pour interroger les informations d'une table de variables API, modifiez le code de programme suivant :

```
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");
}
```

7.18.8.9 Lire la date et l'heure de la dernière modification d'une table de variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Le format de l'horodatage est l'UTC.

Code du programme

Pour lire l'horodatage d'une table de variables API déterminée, modifiez le code de programme suivant :

```
//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;
}
```

7.18.8.10 Supprimer la table des variables API dans un groupe

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code du programme

Pour supprimer une table des variables déterminée dans un groupe, modifiez le code de programme suivant :

```
//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();
    }
}
```

7.18.8.11 Enumérer des variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Code de programme : énumérer des variables API dans des tables de variables

Pour énumérer toutes les variables API contenues dans une table des variables, modifiez le code de programme suivant :

```
//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
    }
}
```

7.18.8.12 Accéder à des variables API

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Le type `PlcTagComposition` représente un ensemble de variables API.

Code de programme : accéder à une variable API précise

Pour accéder à la variable API de votre choix, modifiez le code de programme suivant : Vous avez accès aux attributs suivants :

- Nom (accès en lecture seule)
- Nom du type de données
- Adresse logique
- Commentaire
- ExternalAccessible
- ExternalVisible
- ExternalWritable

```
//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
}
```

Code de programme : créer des variables

Modifiez le code de programme suivant :

```
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);
}
```


Modifiez le code de programme suivant :

```
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);
}
```

Code de programme : supprimer des variables

Modifiez le code de programme suivant :

```
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();
    }
}
```

7.18.8.13 Accéder à des constantes API

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Le type `PlcUserConstantComposition` représente un ensemble de constantes utilisateur API. Vous avez accès aux attributs suivants :

- Nom (accès en lecture seule)
- Nom du type de données
- Valeur

Le type `PlcSystemConstantComposition` représente un ensemble de constantes système API. Vous avez accès aux attributs suivants :

- Nom (accès en lecture seule)
- Nom du type de données (accès en lecture seule)
- Valeur (accès en lecture seule)

Code de programme : créer des constantes utilisateur

Modifiez le code de programme suivant :

```
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);
}
```

Code de programme : supprimer des constantes utilisateur

Modifiez le code de programme suivant :

```
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();
    }
}
```

Code de programme : accéder à des constantes système

Modifiez le code de programme suivant :

```
//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
}
```

Voir aussi

Créer les groupes personnalisés pour variables API (Page 338)

Supprimer les groupes personnalisés pour variables API (Page 339)

Supprimer la table des variables API dans un groupe (Page 342)

Accéder à des variables API (Page 344)

Démarrage de l'éditeur de variables API (Page 335)

Lire la date et l'heure de la dernière modification d'une table de variables API (Page 342)

7.19 Startdrive

7.19.1 Référence

7.19.1.1 DriveObject

DriveObject

La classe `DriveObject` permet d'accéder à l'objet entraînement. L'objet entraînement permet par ex. d'accéder aux paramètres d'entraînement ou au télégramme.

Namespace : `Siemens.Engineering.MC.Drives`

Assembly : `Siemens.Engineering.MC.Drives` dans `Siemens.Engineering.dll`

Le tableau suivant décrit la **syntaxe** de la classe :

```
public sealed class DriveObject
```

Le tableau suivant décrit les **propriétés** de la classe :

Nom	Type de données	Description
Parameters	DriveParameter-Composition (Page 350)	Renvoie une liste des paramètres disponibles de l'objet entraînement.
Telegrams	TelegramComposition (Page 353)	Renvoie une liste des télégrammes disponibles de l'objet entraînement. Cette liste peut être modifiée avec la classe <code>TelegramComposition</code> .

Voir aussi

Déterminer un objet entraînement (Page 361)

7.19.1.2 OnlineDriveObject

OnlineDriveObject

La classe `OnlineDriveObject` permet d'accéder en ligne à l'objet entraînement. L'objet entraînement permet d'accéder aux paramètres d'entraînement.

Namespace : `Siemens.Engineering.MC.Drives`

Assembly : `Siemens.Engineering.MC.Drives` dans `Siemens.Engineering.dll`

Le tableau suivant décrit la **syntaxe** de la classe :

```
public sealed class OnlineDriveObject
```

Le tableau suivant décrit les **propriétés** de la classe :

Nom	Type de données	Description
Parameters	DriveParameter-Composition (Page 350)	Renvoie une liste des paramètres disponibles de l'objet entraînement en ligne. null si le mode est "Hors ligne". En mode hors ligne, l'appel d'une méthode ou l'accès en écriture à un paramètre déclenche une exception.

Voir aussi

Déterminer un objet entraînement (Page 361)

Lire et écrire des paramètres (Page 361)

Lire et écrire des paramètres en ligne (Page 362)

7.19.1.3 DriveObjectContainer

DriveObjectContainer

Le `DriveObjectContainer` est un service de l'objet entraînement (`DeviceItem`) de l'appareil actuel (`Device`).

Le tableau suivant décrit les **navigateurs** du `DriveObjectContainer` :

Nom	Type de données	Description
DriveObjects	DriveObjectComposition	Renvoie une liste des objets entraînement disponibles. Les objets entraînement permettent d'accéder aux paramètres d'entraînement et aux télégrammes.

Voir aussi

DriveParameterComposition (Page 350)

7.19.1.4 OnlineDriveObjectContainer

OnlineDriveObjectContainer

Le `OnlineDriveObjectContainer` est un service de l'objet entraînement (`DeviceItem`) de l'appareil actuel (`Device`).

Le tableau suivant décrit les **navigateurs** du `OnlineDriveObjectContainer` :

Nom	Type de données	Description
<code>OnlineDriveObjects</code>	<code>OnlineDriveObjectComposition</code>	Renvoie une liste des objets entraînement en ligne disponibles (<code>OnlineDriveObject</code> (Page 348)). Les objets entraînement permettent d'accéder aux paramètres d'entraînement.

7.19.1.5 DriveParameterComposition

DriveParameterComposition

La classe `DriveParameterComposition` permet d'accéder aux paramètres de l'entraînement. Tous les paramètres de l'entraînement ne sont pas accessibles via Openness.

Namespace : `Siemens.Engineering.MC.Drives`

Assembly : `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Le tableau suivant décrit la **syntaxe** de la classe :

```
public sealed class DriveParameterComposition
```

Le tableau suivant décrit les **méthodes** de la classe :

Nom	Description
Find(string)	Renvoie l'objet DriveParameter (Page 351) recherché via le nom. null si le paramètre n'est pas trouvé Exemple Find("P108[1]");
Find(UInt16, Int32)	Renvoie l'objet DriveParameter (Page 351) recherché via l'indice de paramètre et de tableau. null si le paramètre n'est pas trouvé Exemples <ul style="list-style-type: none"> cu.Find(108, 1); cu.Find(51, -1);
WriteParameters(IEnumerable<string>, IEnumerable<string>, bool)	Écrit les valeurs dans les paramètres. Avec le réglage ignoreErrors = true, le système tente, en cas d'erreur, d'écrire toutes les valeurs, puis il déclenche une EngineeringTargetInvocationException. Pour les entraînements SINAMICS-G, les valeurs de paramètres ne peuvent être écrites que lorsqu'un Power Module (PM) est configuré. Exemple <pre>List<string> names = new List<string>(); List<string> values = new List<string>(); names.add("p300[0]"); values.add("17"); names.add("p5391[0]"); values.add("20"); cu.WriteParameters(names, values, true);</pre>

7.19.1.6 DriveParameter

DriveParameter

La classe `DriveParameter` permet d'accéder à un paramètre d'entraînement. Tous les paramètres de l'entraînement ne sont pas accessibles via Openness.

Namespace : Siemens.Engineering.MC.Drives

Assembly : Siemens.Engineering.MC.Drives dans
Siemens.Engineering.dll

Le tableau suivant décrit la **syntaxe** de la classe :

```
public sealed class DriveParameter
```

Le tableau suivant décrit les **propriétés** de la classe :

Nom	Type de données	Description
ArrayIndex	Int32	Renvoie l'indice d'un paramètre de tableau. Plage de valeurs : 0-7FFF Pour les paramètres sans tableau, l'indice est -1 Exemple <code>p108[4].15</code> <code>par.ArrayIndex</code> donne 4
ArrayLength	Int32	Renvoie le nombre d'éléments du tableau. Pour les paramètres sans tableau, la valeur est 0
Bits	DriveParameterComposition	Renvoie un objet <code>DriveParameter</code> pour un bit du paramètre. Il est ainsi possible de lire, par exemple, la valeur ou le nom d'un paramètre de bit. Exemple <code>DriveParameter param133 =</code> <code>cu.Parameters.Find(133, 0);</code> <code>DriveParameter param133Bit1 =</code> <code>param133.Bits[1];</code> <code>String paramName = param133Bit1.Name;</code>
EnumValueList	IDictionary<int, string>	Renvoie une liste des valeurs possibles pour le paramètre enum. Par ex. <1, [1] Quick commissioning> null si le paramètre n'est pas de type Enum.
MaxValue	Object	Renvoie la valeur maximale pour l'unité actuellement sélectionnée.
MinValue	Object	Renvoie la valeur minimale pour l'unité actuellement sélectionnée.
Name	string	Renvoie le nom du paramètre. Par ex. "p108[0].2"
ParameterText	string	Renvoie le texte de la description succincte du paramètre.
Number	Int32	Renvoie le numéro du paramètre. Exemple <code>p108[0].2</code> La valeur de retour est 108

Nom	Type de données	Description
Unit	string	Renvoie l'unité du paramètre sous forme de texte.
Value	Object	<p>Renvoie la valeur hors ligne/en ligne du paramètre ou écrit une valeur pour le paramètre.</p> <p>En cas d'erreur d'écriture, une <code>EngineeringTargetInvocationException</code> est déclenchée.</p> <p>Exemples</p> <ul style="list-style-type: none"> • <code>P2080Bit6.Value = 0;</code> • <code>P2080Bit6.Value = cu.Parameters.Find("r19");</code> <p>Source FCOM</p> <p>Le paramètre d'une source FCOM ne peut qu'être lu</p> <p>Puits de signal FCOM</p> <p>Les valeurs possibles sont 0, 1 ou un objet <code>DriveParameter</code>.</p> <p>Un objet <code>DriveParameter</code> est renvoyé lorsque le puits de signal FCOM est lié à un autre paramètre.</p> <p>Voir aussi l'exemple Lire et écrire des paramètres FCOM (Page 363).</p>

Voir aussi

Lire et écrire des paramètres (Page 361)

7.19.1.7 TelegramComposition

TelegramComposition

La classe `TelegramComposition` permet d'accéder aux télégrammes d'un objet entraînement. La structure d'un télégramme peut être lue via la classe `Telegram` (Page 354).

Les télégrammes PROFIsafe ne sont pas pris en charge.

Notez que la classe `TelegramComposition` peut invalider des objets référencés. Par ex., après modification de la taille du télégramme, l'objet `Telegram` (Page 354) devient invalide.

Namespace : `Siemens.Engineering.MC.Drives`

Assembly : `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Le tableau suivant décrit la **syntaxe** de la classe :

```
public sealed class TelegramComposition
```

Le tableau suivant décrit les **méthodes** de la classe :

Nom	Description
CanInsertAdditionalTelegram(Int32, Int32)	Renvoie <code>true</code> si une extension peut être générée conformément aux tailles paramétrées (tailles d'entrée et de sortie).
InsertAdditionalTelegram(Int32, Int32)	Génère, pour l'objet entraînement, une extension conformément aux tailles paramétrées et renvoie <code>true</code> si l'extension a pu être insérée. En cas d'erreur, une <code>EngineeringTargetInvocationException</code> est déclenchée.
CanInsertSupplementaryTelegram(Int32)	Renvoie <code>true</code> si un télégramme supplémentaire peut être généré conformément au numéro de télégramme paramétré.
InsertSupplementaryTelegram(Int32)	Génère le télégramme supplémentaire avec le numéro de télégramme paramétré et renvoie <code>true</code> si le télégramme a pu être inséré. En cas d'erreur, une <code>EngineeringTargetInvocationException</code> est déclenchée.
EraseTelegram(TelegramType)	Renvoie <code>true</code> si le télégramme paramétré a pu être supprimé. Les télégrammes standard ne peuvent pas être supprimés. En cas d'erreur, une <code>EngineeringTargetInvocationException</code> est déclenchée.
Find(TelegramType)	Renvoie l'objet Telegram (Page 354) s'il a pu être trouvé via le type de télégramme paramétré. <code>null</code> si le télégramme n'est pas trouvé. Exemple <code>Telegram telegram = telegrams.Find(TelegramType.MainTelegram);</code>

7.19.1.8 Telegram

Telegram

La classe `Telegram` permet d'accéder à la structure d'un télégramme d'un objet entraînement.

Namespace : `Siemens.Engineering.MC.Drives`

Assembly : `Siemens.Engineering.MC.Drives` dans `Siemens.Engineering.dll`

Le tableau suivant décrit la **syntaxe** de la classe :

```
public sealed class Telegram
```

Le tableau suivant décrit les **propriétés** de la classe :

Nom	Type de données	Description
TelegramNumber	Int32	Renvoie le numéro du télégramme principal ou définit ce numéro. Un télégramme libre porte le numéro 999.
Type	TelegramType (Page 355)	Renvoie le type du télégramme sous la forme Enum TelegramType.
Addresses	AddressComposition (Page 356)	Renvoie une AddressComposition avec des informations sur l'adresse.
PKW	Telegram	Renvoie le canal PKW sous forme de Telegram. null si la propriété n'est pas disponible Un télégramme avec PKW est, par exemple, le télégramme 353.

Le tableau suivant décrit les **méthodes** de la classe :

Nom	Description
CanChangeTelegram(Int32)	Renvoie true si le télégramme peut être modifié dans le type standard paramétré.
GetSize(AddressloType (Page 357))	Renvoie la taille des entrées ou des sorties du télégramme.
CanChangeSize(AddressloType (Page 357), Int32, bool)	Renvoie true si la taille du télégramme peut être modifiée comme elle a été paramétrée. Les télégrammes standard peuvent uniquement être agrandis. La conservation de l'adresse de télégramme actuelle est prise en compte si l'option est paramétrée avec true.
ChangeSize(AddressloType (Page 357), Int32, bool)	Renvoie true si la taille du télégramme a pu être modifiée comme elle a été paramétrée. La conservation de l'adresse de télégramme actuelle est prise en compte si l'option est paramétrée avec true.

7.19.1.9 TelegramType

TelegramType

Enum TelegramType contient des types de télégrammes prédéfinis.

Namespace : Siemens.Engineering.MC.Drives

Assembly : Siemens.Engineering.MC.Drives dans
Siemens.Engineering.dll

Le tableau suivant décrit la **syntaxe** de la classe :

```
public enum TelegramType
```

Le tableau suivant décrit les **entrées Enum** :

Nom	Description
MainTelegram	ID du télégramme principal
SupplementaryTelegram	ID du télégramme supplémentaire
AdditionalTelegram	ID d'une extension

7.19.1.10 AddressComposition

AddressComposition

La classe `AddressComposition` représente l'adresse d'un télégramme.

Namespace : `Siemens.Engineering.MC.Drives`

Assembly : `Siemens.Engineering.MC.Drives` in `Siemens.Engineering.dll`

Le tableau suivant décrit la **syntaxe** de la classe :

```
public sealed class AddressComposition
```

Le tableau suivant décrit les **propriétés** de la classe :

Nom	Type de données	Description
<code>IoType</code>	<code>AddressIoType</code> (Page 357)	Renvoie des informations sur le type de l'adresse.
<code>Context</code>	<code>AddressContext</code> (Page 356)	Renvoie des informations sur le contexte de l'adresse.
<code>StartAddress</code>	<code>Int32</code>	Renvoie l'adresse de début du télégramme ou définit cette adresse.
<code>Length</code>	<code>Int32</code>	Renvoie la longueur du télégramme.

Voir aussi

[TelegramType](#) (Page 355)

7.19.1.11 AddressContext

AddressContext

Enum `AddressContext` contient des informations sur le contexte de l'adresse.

Namespace : `Siemens.Engineering.MC.Drives`

Assembly : `Siemens.Engineering.MC.Drives` dans `Siemens.Engineering.dll`

Le tableau suivant décrit la **syntaxe** de la classe :

```
public enum AddressContext
```

Le tableau suivant décrit les **entrées Enum** :

Nom	Description
AddressContext.None	Un contexte a été trouvé pour l'adresse
AddressContext.Device	Le contexte est une adresse d'appareil
AddressContext.Head	Le contexte est une adresse de tête

7.19.1.12 AddressIoType

AddressIoType

Enum AddressIoType contient des informations sur le type de l'adresse.

Namespace : Siemens.Engineering.MC.Drives

Assembly : Siemens.Engineering.MC.Drives dans
Siemens.Engineering.dll

Le tableau suivant décrit la **syntaxe** de la classe :

```
public enum AddressIoType
```

Le tableau suivant décrit les **entrées Enum** :

Nom	Description
AddressIoType.None	Le type d'IO ne peut pas être utilisé
AddressIoType.Input	Le type est une adresse d'entrée
AddressIoType.Output	Le type est une adresse de sortie
AddressIoType.Diagnosis	Le type est une adresse de diagnostic
AddressIoType.Substitute	Le type est une adresse de remplacement

7.19.1.13 StartDriveDownloadCheckConfiguration

StartDriveDownloadCheckConfiguration

La classe StartDriveDownloadCheckConfiguration est dérivée de la classe DownloadCheckConfiguration et a les mêmes propriétés. La classe DownloadCheckConfiguration est décrite dans l'aide Openness standard.

Cette classe fournit à l'utilisateur les réglages de configuration via des case à cocher.

Le tableau suivant décrit les **propriétés** de la classe :

Nom	Type de données	Description
Checked	bool	Renvoie le réglage actuel de la configuration ou active/désactive la configuration.

Voir aussi

Téléchargement (Page 364)

7.19.2 Exemple de code

Les exemples de code suivants décrivent la procédure de base pour différents cas d'application. Le code n'est pas nécessairement complet et compilable.

7.19.2.1 Créer un groupe d'entraînement

La méthode `CreateWithItem()` de la collection `Devices` permet de créer un groupe d'entraînement.

Les groupes d'entraînement sont spécifiés via les paramètres de la méthode. Le format des paramètres est décrit ci-après.

Créer un groupe d'entraînement G120

Format des paramètres pour G120 :

```
CreateWithItem(@"OrderNumber: mlfb / FirmwareVersion /",
"NameOfTheDevice", positionNumber)
```

L'exemple suivant montre comment créer un groupe d'entraînement G120.

Créer un groupe d'entraînement G120

```
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);
```

Créer un groupe d'entraînement S120, S150, MV, G130, G150

Format des paramètres pour S120, S150, MV, G130, G150 :

```
CreateWithItem(@"OrderNumber: mlfb / FirmwareVersion /
AdditionalTypeIdentifier", "NameOfTheDevice", positionNumber)
```

Valeurs possibles pour `AdditionalTypeIdentifier` :

- Chaîne vide (par ex. pour G120)
- S120

- S150
- MV
- G130
- G150

L'exemple suivant montre comment créer un groupe d'entraînement S120.

Créer un groupe d'entraînement S120

```
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);
```

7.19.2.2 Créer un composant d'entraînement

La méthode `PlugNew()` d'un objet `Device` permet de créer un composant d'entraînement pour un groupe d'entraînement.

L'exemple suivant montre comment créer un composant d'entraînement.

Créer un Motor Module

```
DeviceItem subModul = sdrDevice.PlugNew(@"OrderNumber:6SL3xxx-xxxxx-xxxx",
"MotorModul", 65535);
```

7.19.2.3 Créer un composant pour un composant d'entraînement (uniquement S120)

Pour S120, il est possible de créer un composant sous un composant d'entraînement.

Pour distinguer les codeurs, indiquez en plus une désignation de type. Les désignations de type possibles et les restrictions pour les codeurs sont répertoriées dans le tableau ci-dessous.

L'exemple suivant montre comment créer un composant sous un composant d'entraînement.

Créer un moteur et un codeur sous un Motor Module

```
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);
```

Désignations de type pour les codeurs et restrictions

Lors de l'insertion de codeurs via Openness, les restrictions suivantes s'appliquent :

- Pour certains codeurs, il n'est possible de créer qu'un Sensor Module non spécifié lors de l'insertion via Openness. Dans ce cas, vous devez configurer le type concret de Sensor Module dans TIA Portal.
- Pour un Motor Module, il est possible d'insérer au maximum deux codeurs.

Le tableau suivant dresse la liste des désignations de type disponibles pour les codeurs.

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	En- Dat_2.1.2051
DRIVE-CLIQ. 204	Resolver. 1001	SIN_COS. 2001	SSI.3081	SIN_COS_ +_SSI.2081	HTL_TTL. 3001	HTL_TTL +SSI.3088	En- Dat_2.1.2052
DRIVE-CLIQ. 212	Resolver. 1002	SIN_COS. 2002	SSI.3082	SIN_COS_ +_SSI.2082	HTL_TTL. 3002	HTL_TTL +SSI.3090	En- Dat_2.1.2053
DRIVE-CLIQ. 214	Resolver. 1003	SIN_COS. 2003	SSI.9999	SIN_COS_ +_SSI.2083	HTL_TTL. 3003	HTL_TTL +SSI.9999	En- Dat_2.1.2054
DRIVE-CLIQ. 242	Resolver. 1004	SIN_COS. 2004		SIN_COS_ +_SSI.2084	HTL_TTL. 3005		En- Dat_2.1.2055
DRIVE-CLIQ. 244	Resolver. 9999	SIN_COS. 2005		SIN_COS_ +_SSI.9999	HTL_TTL. 3006		En- Dat_2.1.2151
DRIVE-CLIQ. 9999		SIN_COS. 2006			HTL_TTL. 3007		En- Dat_2.1.9999
DRIVE-CLIQ. 10100		SIN_COS. 2007			HTL_TTL. 3008		En- Dat_2.1.1010 0
		SIN_COS. 2008			HTL_TTL. 3009		
		SIN_COS. 2010			HTL_TTL. 3011		
		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					

7.19.2.4 Déterminer un objet entraînement

Les exemples suivants montrent comment déterminer des objets entraînement hors ligne et en ligne.

Déterminer un objet entraînement hors ligne

```
using Siemens.Engineering.MC.Drives;
//G device
Project project = portal.Projects.Open("..."); //Destination folder to
open the project
DeviceItem item = project.Devices[0].Items[0].Items[0];
item.GetService<DriveObjectContainer>().DriveObjects[0];

//S device
Project project = portal.Projects.Open("..."); //Destination folder to
open the project
DeviceItem item = project.Devices[0].Items[0];
item.GetService<DriveObjectContainer>().DriveObjects[0];
```

Déterminer un objet entraînement en ligne

```
using Siemens.Engineering.MC.Drives;
//G device
Project project = portal.Projects.Open("..."); //Destination folder to
open the project
DeviceItem item = project.Devices[0].Items[0].Items[0];
item.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];

//S device
Project project = portal.Projects.Open("..."); //Destination folder to
open the project
DeviceItem item = project.Devices[0].Items[0];
item.GetService<OnlineDriveObjectContainer>().OnlineDriveObjects[0];
```

7.19.2.5 Lire et écrire des paramètres

L'exemple suivant montre comment lire et écrire des valeurs de paramètres d'entraînement. Pour l'accès, un objet entraînement est nécessaire.

Accès aux paramètres

```
using Siemens.Engineering.MC.Drives;
```

Accès aux paramètres

```
//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;
}

//accessing bit value
DriveParameter parameter133 = driveObject.Parameters.Find(133,0);
if (parameter133 != null)
{
    DriveParameter parameter133Bit1 = parameter133.Bits[1];
    if (parameter133Bit1 != null)
    {
        Console.WriteLine("The name of the parameter is : " +
parameter133Bit1.Name);
        Console.WriteLine("The value of the second bit of the parameter is : "
+ parameter133Bit1.Value.ToString());
    }
}

//Get the enum values of a parameter
DriveParameter parameter10 = driveObject.Parameters.Find("p10");
foreach (var enumItem in parameter10.EnumValueList)
{
    Console.WriteLine("Enum value: " + enumItem.Key.ToString() + " = " +
enumItem.Value);
}

```

7.19.2.6 Lire et écrire des paramètres en ligne

L'exemple suivant montre comment recevoir une liste des paramètres en ligne disponibles. Pour l'accès, un objet entraînement en ligne est nécessaire.

L'accès en lecture et en écriture à des paramètres en ligne individuels à partir de la liste des paramètres est identique à l'exemple Lire et écrire des paramètres (Page 361).

Accès aux paramètres en ligne

```
using Siemens.Engineering.MC.Drives;
```

Accès aux paramètres en ligne

```

OnlineDriveObject onlineDo =
((IEngineeringServiceProvider)item).GetService<OnlineDriveObjectContainer>
().OnlineDriveObjects[0];
if (onlineDo != null)
{
    var parameters = onlineDo.Parameters;
}

```

7.19.2.7 Lire et écrire des paramètres FCOM

L'exemple suivant montre comment lire et écrire des valeurs de paramètres FCOM. Pour l'accès, un objet entraînement est nécessaire.

Lire des paramètres FCOM

```

using Siemens.Engineering.MC.Drives;
DriveParameter bicoSink= driveObject.Parameters.Find("p681");
if(bicoSink!=null)
{
    if(bicoSink.Value is DriveParameter)
    {
        DriveParamter bicoSourceValue = bicoSink.Value as DriveParameter;
        Console.WriteLine("The value of " + bicosink.Name + "parameter: " +
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 " + bicosink.Name + "parameter: " +
bicoSink.Value.ToString());
    }
}

```

Écrire des paramètres FCOM

```

using Siemens.Engineering.MC.Drives;
DriveParameter bicoSource= driveObject.Parameters.Find("r19");
if(bicoSource != null)
{
    try
    {
        bicoSink.Value = bicoSource;
    }
    catch(UserException ex)
    {
        Console.WriteLine("Write failure :" + ex.Message);
    }
}

```

7.19.2.8 Insérer et étendre des télégrammes

L'exemple suivant montre comment insérer une extension et modifier la taille d'un télégramme standard. Pour l'accès, un objet entraînement est nécessaire.

Insérer une extension et modifier la taille d'un télégramme standard

```
using Siemens.Engineering.MC.Drives;
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)
}
```

7.19.2.9 Téléchargement

Après le lancement du téléchargement, l'utilisateur doit adapter ou confirmer les réglages de configuration. Les réglages de configuration sont fournis en tant qu'objets enfants de l'objet `DownloadConfiguration` et disponibles dans trois types distincts :

- `StartDriveDownloadCheckConfiguration`
- `DownloadSelectionConfiguration`
- `DownloadPasswordConfiguration`

L'exemple suivant montre l'évaluation des différents types de réglages de configuration dans PreDownload Delegate.

Évaluation des réglages de configuration après lancement du téléchargement

```
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;
    }
}
```

L'exemple suivant montre comment charger un projet dans l'appareil.

Téléchargement dans l'appareil S120

```
using Siemens.Engineering.Download;
using Siemens.Engineering.Online;
```

Téléchargement dans l'appareil 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);
}
```

7.20 Exceptions

7.20.1 Traitement des exceptions

Exceptions en cas d'accès à TIA Portal via des API TIA Portal Openness

Lors de l'exécution d'une application TIA Portal Openness avec la TIA Portal Openness API, toutes les erreurs qui se sont produites sont signalées comme des exceptions. Ces exceptions contiennent des informations qui vous aident à éliminer les erreurs survenues.

Il existe deux types d'exception :

- Recoverable (Siemens.Engineering.EngineeringException)
Avec cette exception, vous continuez d'accéder à TIA Portal sans interruption. Vous pouvez également couper la liaison au portail TIA.

Les EngineeringExceptions contiennent les types suivants :

- Exceptions de sécurité (EngineeringSecurityException), par exemple en cas d'absence de droits d'accès.
- Exceptions lors de l'accès aux objets (EngineeringObjectDisposedException), par ex. lors de l'accès à des objets qui n'existent plus.
- Exceptions lors de l'accès aux attribut (EngineeringNotSupportedException), par ex. lors de l'accès à des attributs qui n'existent plus.
- Exceptions générales lors de l'appel (EngineeringTargetInvocationException), par ex. en cas d'erreur malgré des appels valides de la TIA Portal Openness API.
- Exceptions lors de l'appel (EngineeringRuntimeException), par ex. lors d'une affectation invalide.
- Exceptions si les ressources sont insuffisantes dans l'instance de TIA Portal affectée (EngineeringOutOfMemoryException)
- Exceptions lorsque les appels sont terminés (EngineeringUserAbortException), par ex. lors de l'interruption du processus d'importation par l'utilisateur.
- Exceptions lors de l'appel de l'API par un délégué mis à disposition par un client (EngineeringDelegateInvocationException). Cette exception est dérivée de l'exception EngineeringTargetInvocationException.

Les EngineeringExceptions ont les attributs suivants :

- `ExceptionMessageData messageData` : Contient la cause pour laquelle l'exception a été déclenchée.
- `ExceptionMessageData detailMessageData` : Contient des informations supplémentaires sur la cause. Le résultat est fourni en retour sous forme de <IList>.
- `String message` : Fournit en retour le résultat issu de `MessageData` et de `DetailMessageData`.

`ExceptionMessageData` fournit en retour les informations suivantes :

- `String Text` : Contient la cause pour laquelle l'exception a été déclenchée.
 - `Int ServiceId` : Fournit l'ID du service ayant déclenché l'exception.
 - `Int MessageId` : ID univoque au sein du service.
- NonRecoverable (Siemens.Engineering.NonRecoverableException)
Dans le cas de cette exception, le portail TIA est fermé et la liaison au portail TIA est coupée. Vous devez redémarrer le TIA Portal avec l'application TIA Portal Openness.

Code de programme

L'exemple suivant montre les possibilités que vous avez pour réagir à des exceptions :

```
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);
}
```


Exportation/importation

8.1 Vue d'ensemble

8.1.1 Notions élémentaires sur l'importation/exportation

Introduction

Vous pouvez exporter certaines données de configuration puis les réimporter après édition, soit dans le même projet, soit dans un autre.

Remarque

L'utilisation de cette description pour éditer et exploiter manuellement le fichier source n'entraîne aucune obligation ni garantie d'aucune sorte. Siemens décline donc toute responsabilité en cas d'utilisation de cette description ou de parties de cette description.

Objets exportables et importables

Vous pouvez également importer ou exporter les données de configuration suivantes par le biais de la TIA Portal Openness API :

Tableau 8-1 Projets

Objets	Exportation	Importation
Bibliothèque de graphiques	X	X

Tableau 8-2 API

Objets	Exportation	Importation
Blocs	X	X
Blocs avec protection Know How	X	-
Blocs F	X	-
Blocs système	X	-
Tables de variables API	X	X
Variables et constantes API	X	X
Types de données utilisateur	X	X

Tableau 8-3 IHM

Objets	Exportation	Importation
Vues	X	X
Modèles de vue	X	X
Vues globales	X	X
Vues contextuelles	X	X
Vues Slide-in	X	X
Scripts	X	X
Listes de textes	X	X
Listes de graphiques	X	X
Cycles	X	X
Connexions	X	X
Table des variables	X	X
Variables	X	X

Exportation complète ou de références ouvertes

Les types d'objets dont la liste est dressée ci-dessus sont exportés ou importés avec tous les objets lorsqu'ils appartiennent à la même arborescence. Cela vaut également pour les objets référencés de la même arborescence.

Les objets référencés dans d'autres arborescences ne peuvent pas, quant à eux contraire, être complètement exportés ou importés. Des "références ouvertes" à ces objets sont exportées ou importées à leur place.

Les objets référencés de la même arborescence sont exportés uniquement s'ils font partie du groupe des objets exportables. Toutes les dynamisations s'appliquant à des objets sont traitées comme des objets lors de l'importation/exportation et sont également exportées et importées.

Lors de l'exportation, tous les attributs d'objet qui ont été modifiés durant la configuration sont exportés. Cela s'applique toujours, qu'un attribut modifié soit utilisé ou non.

Exemple : Vous avez configuré un champ d'E/S graphique avec le mode "Entrée/Sortie" et sélectionné le réglage "Visible après avoir cliqué" pour l'attribut "Barre de défilement". Puis vous avez basculé le mode sur "Deux états" pendant la configuration. Dans ce mode, l'attribut "Barre de défilement" n'est pas disponible. Étant donné que l'attribut "Barre de défilement" a été modifié, il est exporté lors de l'exportation, bien qu'il ne soit pas utilisé.

Importation de références ouvertes

Vous pouvez également importer des objets assortis de références ouvertes (voir Importation de données de configuration (Page 377)).

Si les objets référencés se situent dans le projet cible, les références ouvertes sont automatiquement liées à nouveau aux types d'objet. Ces objets doivent se situer au même endroit et porter le même nom que pour l'exportation. Si les objets référencés ne sont pas

situés dans le projet cible, les références ouvertes ne peuvent pas être résolues. Aucun objet supplémentaire n'est créé pour la résolution des références ouvertes.

Importation et exportation du format de fichier

Le format de fichier à exporter et à importer est XML. Le format AML est requis uniquement pour les données CAx. Vous trouverez les différentes définitions de schéma pour tous les formats dans le chapitre correspondant du présent manuel :

- Format XML pour les données d'un appareil IHM (Page 385)
- Format XML pour les données d'un appareil API (Page 436)
- Format AML pour les données CAx (Page 503)

Importation et exportation de polices de caractère

Les polices définies pour des objets sont également exportées et importées.

Si vous importez des polices qui ne sont pas incluses dans le projet, la police par défaut s'affiche pour l'objet après l'importation. La police importée est toutefois enregistrée dans la gestion des données.

Si les attributs d'une police ne sont pas définis dans le fichier d'importation, les attributs sont dotés de valeurs par défaut après l'importation.

Restrictions

Le format d'exportation est interne et n'est valable que pour la version actuelle de TIA Portal Openness. Le format d'exportation peut être modifié pour les versions ultérieures.

Toutes les erreurs survenant au cours de l'importation ou de l'exportation sont signalées comme des exceptions.

Pour plus d'informations sur les exceptions, veuillez vous référer au chapitre Traitement des exceptions (Page 367).

Voir aussi

Domaine d'utilisation de l'importation/exportation (Page 373)

Exportation de données de configuration (Page 375)

8.1.2 Domaine d'utilisation de l'importation/exportation

Introduction

La fonction d'importation/exportation vous permet d'exporter certains objets de manière ciblée.

Vous pouvez éditer les données exportées avec un programme externe ou les réutiliser telles quelles dans d'autres projets TIA Portal.

Si vous structurez correctement le fichier d'importation, vous pouvez également importer sans exportation préalable des données de configuration créées en externe.

Remarque

L'importation de données de configuration créées en externe avec des erreurs de code ou de structure erronée peut provoquer des erreurs inattendues.

Domaine d'application

Exporter et importer des données est utile pour les tâches suivantes :

- éditer des données de configuration en externe,
- importer des données de configuration générées en externe, telles que des listes de textes et des variables,
- distribuer à différents projets des données de configuration prédéfinies, p. ex. une vue de processus modifiée qui doit être utilisée dans plusieurs projets.
- Pour la réplique et l'adaptation de la configuration matérielle entre le projet TIA Portal et un programme ECAD.

Voir aussi

Notions élémentaires sur l'importation/exportation (Page 371)

8.1.3 Importation SimaticML spécifique à la version

Utilisation

L'importation SimaticML est utilisable avec toutes les versions à partir de TIA Portal Openness V14 SP1. Vous pouvez importer vos anciens fichiers d'exportation au moins dans les deux versions supérieures.

Pour prendre en charge cette caractéristique, les fichiers SimaticML contiennent maintenant les informations de version de modèle représentées ci-dessous :

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
  <Engineering version="V14 SP1"/>
  <DocumentInfo>
    ...
  </DocumentInfo>
  <SW.DataBlock ID="0">
    ...
  </SW.DataBlock>
</Document>
```

Remarque

Si ces informations de version ne sont pas présentes dans le fichier SimaticML, le système utilise la version de modèle actuelle.

8.1.4 Edition du fichier XML

Introduction

Pour éditer un fichier XML destiné à l'importation de données de configuration, vous utilisez un éditeur XML ou un éditeur de texte.

Si vous effectuez des modifications importantes ou si vous créez vous-même des structures d'objet, il est recommandé d'utiliser un éditeur XML disposant d'une fonction de complément automatique.

Remarque

La modification du contenu XML requiert de solides connaissances de la structure et des règles de validation dans XML. Évitez les erreurs de validation et ne modifiez manuellement la structure XML qu'exceptionnellement.

8.1.5 Exportation de données de configuration

Introduction

Les données de configuration sont à chaque fois exportées dans un fichier XML par objet de départ (racine).

L'édition du fichier d'exportation requiert des connaissances en XML. Pour une édition simplifiée, utilisez un éditeur XML.

Exemple

Vous avez une vue de processus qui contient un champ E/S. Une variable externe est configurée pour ce champ E/S. Si vous exportez la vue de processus, la vue et le champ E/S sont exportés. La variable et la liaison utilisée par la variable ne sont pas exportées, seule une référence ouverte est exportée.

Contenu du fichier d'exportation

À partir de l'objet de départ, tous les objets d'une arborescence sont stockés, ainsi que leurs attributs. En revanche, toutes les références aux objets d'autres arborescences sont exportés comme références ouvertes uniquement. Les attributs correspondants des objets référencés dans différentes arborescences ne sont pas écrits dans le fichier d'exportation.

Remarque

L'exportation de types d'objet de la bibliothèque n'est pas prise en charge.

Vous pouvez créer des objets comme type dans la bibliothèque. Les instances du type d'objet utilisées dans le projet peuvent être éditées avec l'application TIA Portal Openness comme d'autres objets. Si vous exportez des objets, les instances sont exportées sans les informations de type.

Si vous réimportez ces objets dans le projet, les instances des types d'objet sont écrasées et l'instance est coupée du type d'objet.

Le fichier d'exportation ne contient pas nécessairement tous les attributs d'un objet. C'est vous qui définissez les données à exporter :

- `ExportOptions.None`
Ce paramétrage n'exporte que les données modifiées ou différentes des données standard. Le fichier d'exportation contient, de plus, toutes les valeurs obligatoires pour une importation ultérieure des données.
- `ExportOptions.WithDefaults`¹
De plus, les valeurs par défaut sont exportées.
- `ExportOptions.WithReadOnly`¹
De plus, les valeurs protégées en écriture sont exportées.

¹ : vous pouvez combiner ces deux options avec la syntaxe suivante :

```
Export (path, ExportOptions.WithDefaults |  
ExportOptions.WithReadOnly) ;
```

Le contenu du fichier d'exportation est entièrement en anglais. Indépendamment de cela, les textes de projet sont exportés et importés dans toutes les langues disponibles.

Dans le fichier d'exportation, les données de configuration sont toutes structurées comme objets XML.

Voir aussi

Notions élémentaires sur l'importation/exportation (Page 371)

Exporter des blocs (Page 448)

8.1.6 Importation de données de configuration

Introduction

Les données de configuration sont importées depuis un fichier XML exporté au préalable et édité ou bien depuis un fichier XML que vous créez vous-même. Les données contenues dans ce fichier sont contrôlées lors de l'importation. Cela garantit que l'importation ne provoquera pas une incohérence des données de configuration dans TIA Portal.

Restrictions

- Tous les objets racine dans le fichier d'importation doivent être du même type, par ex. tables de variables, blocs, etc.
- Si plusieurs objets racine sont indiqués dans un fichier d'importation et que l'un de ces objets n'est pas valide, le contenu du fichier d'importation n'est pas importé en entier.
- Lors de l'importation de textes, les langues du projet correspondantes doivent être paramétrées dans le projet cible pour éviter que l'importation n'échoue. Si nécessaire, vous pouvez modifier les paramètres linguistiques via TIA Portal Openness.
- Si vous indiquez, dans le fichier d'importation, des attributs d'un objet invalides non éditables dans l'interface utilisateur graphique de TIA Portal, l'importation est annulée.
- Seul les pointeurs de zone sous "separately for each connection" peuvent être importés ou exportés.
- L'importation de types d'objet de la bibliothèque n'est pas prise en charge. Vous pouvez créer des objets comme type dans la bibliothèque. Les instances du type d'objet utilisées dans le projet peuvent être éditées avec l'application TIA Portal Openness comme d'autres objets. Si vous exportez des objets, les instances sont exportées sans les informations de type. Si vous réimportez ces objets dans le projet, les instances des types d'objet sont écrasées et l'instance est coupée du type d'objet.
- L'importation de blocs de sécurité n'est pas prise en charge.

Remarque

Plages de valeurs pour les attributs graphiques en fonction de l'appareil

Si les valeurs d'attributs graphiques se situent en dehors de la plage de valeurs valide, ces valeurs sont remises aux valeurs maximales possibles pour l'appareil IHM lors de l'importation.

Comportement d'importation différent

Si les objets à importer existent déjà dans le projet, vous devez commander le comportement d'importation à l'aide de différents codes de programme. Faute de quoi, les objets sont de nouveau créés dans le projet lors de l'importation.

Les paramétrages suivants peuvent être effectués pour définir le comportement d'importation :

- `ImportOptions.None`
Ce paramètre permet d'importer les données de configuration sans écrasement. Si un objet existe déjà dans le projet lors de l'importation depuis un fichier XML, le processus est annulé par une exception.
- `ImportOptions.Override`
Ce paramètre est utilisé pour l'importation des données de configuration avec écrasement automatique. Vous pouvez décider d'écraser les objets existants au sein du projet pendant l'importation. Les objets pertinents sont supprimés du projet avant l'importation et recréés avec des valeurs par défaut. Lors de l'importation, ces valeurs par défaut sont écrasées par des valeurs issues de l'importation. Si l'objet existant et le nouvel objet ne sont pas dans le même groupe, ces valeurs ne peuvent pas être écrasées. Pour éviter des conflits de noms, l'importation est annulée et une exception est générée.

Marche à suivre pour l'importation

Pour importer un fichier XML, il faut que les données qu'il contient satisfassent à certaines règles. Le contenu du fichier d'importation doit avoir la forme correcte. Il ne doit présenter aucune erreur de syntaxe ni aucune erreur dans la structure des données. En cas de modifications importantes, utilisez un éditeur XML, car ce dernier contrôle ces critères avant l'importation.

Lors de l'importation du fichier XML dans TIA Portal, les données contenues dans le fichier sont d'abord contrôlées afin d'exclure toute erreur formelle dans le code XML. Si des erreurs sont détectées lors de la vérification, l'importation est interrompue et les erreurs s'affichent dans une exception (voir Traitement des exceptions (Page 367)).

Voir aussi

Notions élémentaires sur l'importation/exportation (Page 371)

Importer un type de données utilisateur (Page 500)

8.2 Importation/exportation de données du projet

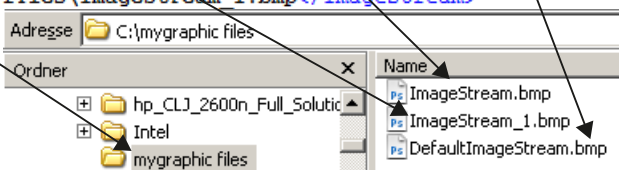
8.2.1 Bibliothèque de graphiques

8.2.1.1 Exportation/importation de graphiques

Introduction

L'exportation de données de configuration de TIA Portal vers le fichier XML ne contient pas de graphique sélectionné ni de graphique référencé par un objet. Ils sont enregistrés séparément lors de l'exportation. Dans le fichier XML, les graphiques sont référencés avec un chemin relatif et le nom de fichier. Dans le fichier XML, une référence à un graphique est structurée comme objet et contient, comme les autres objets, une liste d'attributs ainsi qu'une liste de liens le cas échéant.

```
<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>
```



Exporter des graphiques

L'exportation des données de configuration contient uniquement les graphiques qui ont été directement sélectionnés pour l'exportation. Les graphiques pouvant être exportés sont enregistrés dans TIA Portal pour la langue considérée. Lorsqu'un projet est configuré dans plusieurs langues, toutes les versions de langue utilisées sont exportées.

8.2 Importation/exportation de données du projet

Lors de l'exportation de graphiques, un nouveau dossier est créé dans le dossier du fichier d'exportation. Le nom du dossier est construit en connectant le nom du fichier XML avec des "fichiers". Ce dossier contient les graphiques exportés. Si ce dossier existe déjà, un nouveau dossier est créé, dont le nom est complété avec un numéro d'ordre.

Les graphiques sont enregistrés dans le même format de fichier que dans le projet. Le format n'est ni modifié, ni converti, et la résolution ainsi que la profondeur de couleur restent également inchangées.

L'identifiant "default" est utilisé comme extension de fichier pour la langue sélectionnée comme langue par défaut.

Si le dossier contient déjà un fichier du même nom, le nom de fichier du graphique exporté est complété par un numéro d'ordre.

Importer des graphiques

Les conditions pour l'importation de graphiques sont les suivantes :

- Les graphiques doivent avoir un format de fichier pris en charge par TIA Portal.
- Dans le fichier XML, les graphiques doivent être référencés avec un chemin relatif.

Après l'exportation d'un graphique, celui-ci peut être édité à l'aide d'un programme graphique en dehors de TIA Portal puis réimporté.

Voir aussi

Notions élémentaires sur l'importation/exportation (Page 371)

8.2.1.2 Exporter les graphiques d'un projet

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Vous avez le choix entre exporter un graphique individuel ou exporter tous les graphiques de la bibliothèque de graphiques d'un projet dans toutes les langues. Un fichier XML contenant toutes les entrées du graphique du projet concernées est créé lors de l'exportation et référencé avec les graphiques exportés. Les graphiques concernés sont stockés avec le fichier XML dans le même répertoire du système de fichiers.

Pour que les graphiques exportés (*.jpg, *.bmp, *.png, *.ico etc.) puissent être modifiés, ces graphiques ne sont pas protégés en écriture.

Code du programme : Exporter un graphique

Pour exporter le graphique requis, utilisez le code de programme suivant :

```
//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);
```

Code du programme : Exporter tous les graphiques

Pour exporter tous les graphiques de la bibliothèque de graphiques, modifiez le code de programme suivant :

```
//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);
}
```

8.2.1.3 Importer des graphiques dans un projet

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Un fichier XML est stocké avec les différentes versions linguistiques d'un graphique dans un répertoire de votre système de fichiers.

Vous pouvez référencer tous les graphiques dans un chemin relatif de votre fichier XML.

Vous pouvez désormais importer toutes les versions linguistiques d'un graphique contenu dans le fichier XML dans la bibliothèque de graphiques.

Veillez également tenir compte de ce qui suit Importation de données de configuration (Page 377).

Code du programme

Pour importer un ou plusieurs graphiques, modifiez le code de programme suivant :

```
//Import all language variants of a single graphic
Project project = ...;
MultiLingualGraphicComposition graphicComposition = project.Graphics;
graphicComposition.Import(new FileInfo(@"D:\Graphics\Graphic1.xml"),
ImportOptions.Override);
```

8.2.2 Textes du projet

8.2.2.1 Exportation de textes de projet

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Dans TIA Portal, les textes de projet se trouvent sous le nœud "Langues et ressources" d'un projet. Ces textes sont exportés dans un fichier *.xlsx qui peut, par exemple, être utilisé pour des traductions. Les restrictions valables pour l'interface utilisateur s'appliquent également à l'exportation et l'importation de textes de projet. Restrictions valables :

- Les textes exportés peuvent être importés uniquement dans le projet dont ils ont été exportés.
- Les textes ne peuvent être traduits que dans les langues existant dans le projet. Si nécessaire, vous pouvez ajouter des langues du projet via TIA Portal Openness.
- Seuls les textes existants peuvent être réimportés. Une fois que des textes ont été supprimés du projet d'origine ou créés à nouveau, l'importation de ces textes échoue.

Vous devez définir les paramètres suivants :

Nom	Exemple	Description
pah	new FileInfo("D:\Test\Project-Text.xlsx")	Chemin du fichier d'exportation
sourceLanguage	new CultureInfo("en-US")	Langue de référence de laquelle le texte doit être traduit
targetLanguage	new CultureInfo("de-DE")	Langue de référence dans laquelle le texte doit être traduit

Remarque

Les textes multilingues sont exportés avec l'objet de niveau supérieur auquel ils appartiennent. Les textes multilingues ne peuvent pas être exportés de manière explicite.

Code de programme : exporter depuis le nœud "Langues et ressources"

Des paramètres de l'exemple utilisé, il résulte le code de programme suivant pour l'exportation de textes de projet :

```
project.ExportProjectTexts(new FileInfo(@"D:\Test\ProjectText.xlsx"), new CultureInfo("en-US"), new CultureInfo("de-DE"));
```

Structure XML d'un élément de texte multilingue exporté

```
...
<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>
...
```

8.2.2.2 Importation de textes de projet**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Dans TIA Portal, les textes de projet se trouvent sous le nœud "Langues et ressources" d'un projet. Vous pouvez importer des textes de projet d'un fichier *.xlsx, ce qui peut servir à des fins de traduction, par exemple. Les restrictions valables pour l'interface utilisateur s'appliquent également à l'exportation et l'importation de textes de projet. Restrictions valables :

- Les textes exportés peuvent être importés uniquement dans le projet dont ils ont été exportés.
- Les textes traduits peuvent être importés uniquement dans les langues disponibles dans le projet dont ils ont été exportés.
- Seuls les textes existants peuvent être réimportés. Une fois que des textes ont été supprimés du projet d'origine ou créés à nouveau, l'importation de ces textes échoue.

Vous devez définir les paramètres suivants :

Nom	Exemple	Description
path	new FileInfo(@"D:\Test\Project-Text.xlsx")	Chemin du fichier d'importation
updateSourceLanguage	true	Pour "true", le texte de la langue de référence est actualisé à l'aide du fichier d'exportation. Pour "false", le texte de la langue de référence n'est pas actualisé.

Remarque

Les textes multilingues sont importés avec l'objet de niveau supérieur auquel ils appartiennent. Les textes multilingues ne peuvent pas être importés de manière explicite.

Code de programme

Des paramètres de l'exemple utilisé, il résulte le code de programme suivant pour l'importation de textes de projet :

```
ProjectTextResult result = project.ImportProjectTexts(new FileInfo(@"D:\Test\nProjectText.xlsx"), true);
```

L'importation de textes de projet fournit en retour un objet qui affiche l'état de l'importation et indique le chemin où le journal d'importation est enregistré. Pour accéder à ces attributs, vous pouvez utiliser les codes suivants :

```
ProjectTextResultState resultState = result.State;  
FileInfo logFilePath = result.Path;
```


8.3 Importation/exportation de données d'un appareil IHM

8.3.1 Structure d'un fichier XML

Introduction

Les données du fichier d'exportation issues de l'importation/exportation sont organisées au moyen d'une structure de base.

Structure de base d'un fichier d'exportation

Le fichier d'exportation est créé au format XML.

Le fichier XML commence par des informations sur le document. Il comporte les données de l'installation spécifique à l'ordinateur avec laquelle le projet a été exporté.

Le fichier d'exportation comprend les deux zones suivantes :

- Informations sur le document
Cette zone vous permet d'indiquer vos propres informations relatives à l'exportation et ce dans une syntaxe XML valide. L'importation ignore le contenu.
Vous pouvez par ex. insérer un bloc `<IntegrityInformation>...</IntegrityInformation>` en plaçant des informations supplémentaires à la validation. Après la transmission du fichier XML, le destinataire peut vérifier avant l'importation avec ce bloc si le fichier XML a été modifié.
- Objet
Cette zone contient les éléments à exporter.

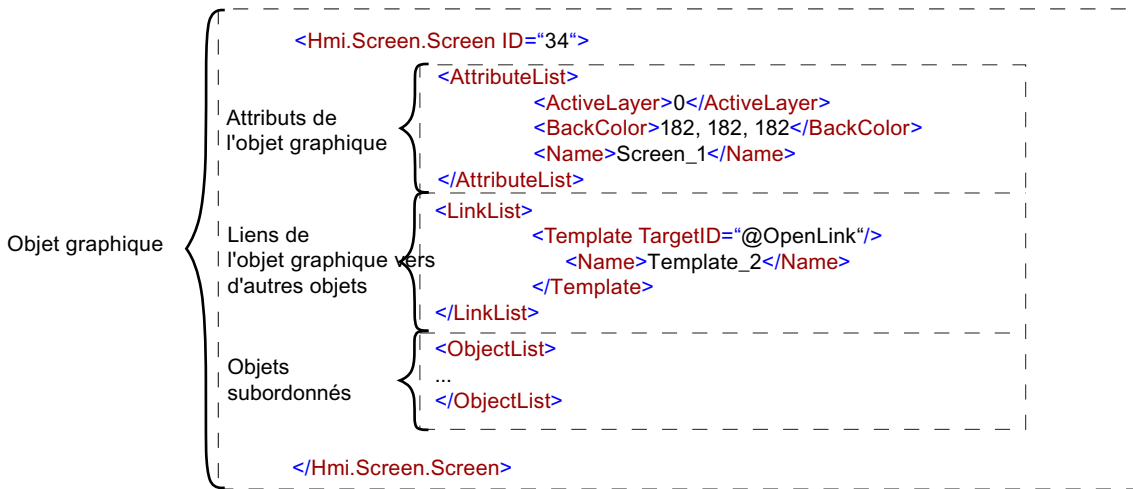
```
<?xml version="1.0" encoding="UTF-8" ?>
<Document xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <DocumentInfo>
    <UserName>Jane Doe</UserName>
    <Company>Example_Inc</Company>
    <IntegrityInformation>...</IntegrityInformation>
    <Created>2016-04-28T18:05:42.179207Z</Created>
    <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>
```

Informations sur le document

Objet graphique

Objets graphiques d'un fichier d'exportation

Les éléments exportés sont disponibles dans d'autres éléments du fichier XML.



Voir aussi

Notions élémentaires sur l'importation/exportation (Page 371)

8.3.2 Structure des données pour l'importation/exportation

Objets

La structure de base est la même pour tous les objets.

Chaque objet du fichier XML débute par son type, p. ex. "Hmi.Screen.Button" et un ID. L'ID est automatiquement générée durant l'exportation.

```
<Hmi.Screen.Button CompositionName="ScreenItems" ID="60">
```

Excepté l'objet de départ, chaque objet contient également un attribut XML "CompositionName". La valeur de cet attribut est prédéfinie. Dans quelques cas, vous devez spécifier cet attribut, p. ex. pour changer d'inscription quand un bouton est enfoncé ou relâché.

```

<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>

```

Attributs

Chaque objet comprend des attributs qui sont contenus dans une section appelée "AttributeList". Chaque attribut est structuré comme élément XML, p. ex. "BackColor". La valeur d'un attribut est structurée comme contenu XML, p. ex "204, 204, 204".

```

<Hmi.Screen.Button ID="2" CompositionName="ScreenItems">
  <AttributeList>
    <BackColor>204, 204, 204</BackColor>
    <ObjectName>Button_1</ObjectName>
  </AttributeList>
</Hmi.Screen.Button>

```

Pour référencer des objets, chaque objet reçoit au besoin une section appelée "LinkList". Cette section contient des liaisons à d'autres objets à l'intérieur ou à l'extérieur du fichier XML. Chaque liaison est structurée comme élément XML. La désignation d'une liaison est prédéfinie par l'objet cible dans le fichier modèle. Chaque liaison comprend également l'attribut "TargetID". Si l'objet cible se trouve dans le fichier XML, la valeur de l'attribut "TargetID" est l'ID de l'objet référencé, précédé de dièse "#". Si l'objet cible ne se trouve pas dans le fichier XML, la valeur de l'attribut "TargetID" est égale à "@OpenLink". La référence à l'objet proprement dite est structurée comme un élément XML subordonné.

```

<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>

```

Corrélation entre les objets et la structure XML

Les figures ci-dessous montrent la corrélation entre la structure XML exportée et les objets correspondants dans WinCC.

```

<Hmi.Screen.Screen ID="34">
  <AttributeList>
    <ActiveLayer>0</ActiveLayer>
    <BackColor>182, 182, 182</BackColor>
    <Name>Screen_1</Name>
  </AttributeList>
  <LinkList>
    <Template TargetID="@OpenLink">
      <Name>Template_2</Name>
    </Template>
  </LinkList>
  <ObjectList>
    <Hmi.Screen.ScreenLayer ID="1" CompositionName="Layers">
      <AttributeList>
        <Index>0</Index>
        <Name>Layer_0</Name>
      </AttributeList>
    </ObjectList>
  </ObjectList>

```

Screen_1 [Screen] Properties

Property list: General, Appearance, Layout, Layers, Security, Miscellaneous, Hide/Show, Design, Web access

ES	RT	Layer	ES	RT	Layer
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_16
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_17
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_18
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_3	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_19
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_4	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_20
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_5	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_21
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_6	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_22
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_7	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_23
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_8	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Layer_24

Settings: Show all ES layers: Active layer: 0

Figure 8-1 Corrélation entre l'interface utilisateur WinCC et la structure XML

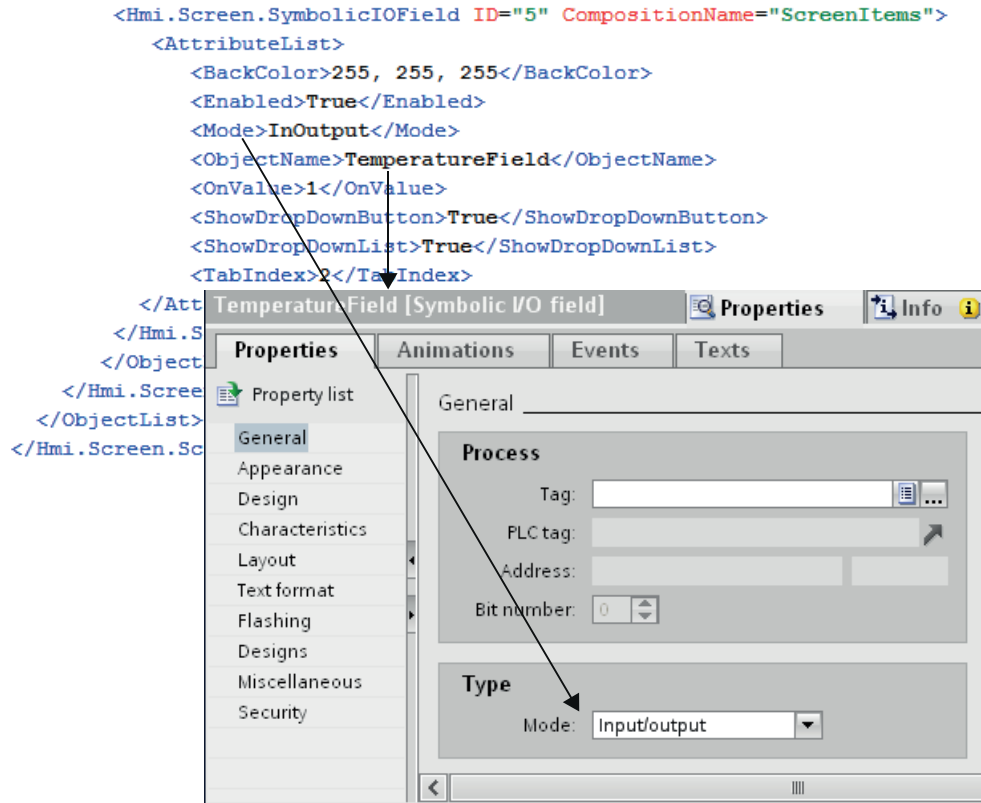


Figure 8-2 Corrélation entre les paramètres de WinCC et la structure XML

8.3.3 Cycles

8.3.3.1 Exportation de cycles

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal. Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert. Voir Ouvrir un projet (Page 99)

Utilisation

L'interface TIA Portal Openness API prend en charge l'exportation de tous les cycles d'un appareil IHM connu vers un fichier XML. La génération du fichier d'export correspondant indique que l'export est terminé.

Code du programme

Pour exporter des cycles d'un appareil IHM vers un fichier XML, modifiez le code de programme suivant :

```
//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);
    }
}
```

8.3.3.2 Importer des cycles

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Si vous utilisez `ImportOptions.None`, le numéro de la composition (Composition count) vous permet de détecter les cycles effectivement importés. Vous avez accès à ces cycles importés.

Remarque

Les cycles standard ont des attributs qui ne peuvent être édités dans l'interface utilisateur. Si vous indiquez dans le fichier d'importation que ces attributs doivent être modifiés, l'importation déclenche une `NonRecoverableException` et ferme TIA Portal.

Code du programme

Pour importer un cycle ou plusieurs cycles dans un appareil IHM depuis un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

Voir aussi

Importation de données de configuration (Page 377)

8.3.4 Table des variables

8.3.4.1 Exporter des tables de variables IHM

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Utilisation

Un fichier XML est exporté par table de variables IHM. L'API prend en charge ce processus d'exportation. L'exportation de tables de variables est aussi disponible dans les sous-dossiers.

Code du programme : Exporter toutes les tables de variables IHM à partir d'un dossier indiqué

Pour exporter toutes les tables de variables IHM d'un dossier défini, modifiez le code de programme suivant :

```
//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);
    }
}
```

Code du programme : Exporter une table de variables IHM

Pour exporter une seule table de variables IHM, modifiez le code de programme suivant :

```
//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);
    }
}
```

Code du programme : Exporter toutes les tables de variables IHM

Pour exporter toutes les tables de variables IHM, modifiez le code de programme suivant :

```
//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);
    }
}
}
```

8.3.4.2 Importer une table de variables IHM

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Code du programme

Pour importer la table de variables IHM d'un fichier XML dans un dossier personnalisé ou un dossier système, modifiez le code de programme suivant :

```
//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);
}
```

Importation erronée de variables

Si vous utilisez les caractères spéciaux suivants dans des noms de variables ou de variables référencées, l'importation de variables échoue :

- . (Point)
- \ (Barre oblique inversée)

Solution 1 :

Vérifiez avant une exportation que les noms des variables à exporter ou des variables référencées ne contiennent aucun point ou barre oblique inversée.

Solution 2 :

Ajoutez dans le fichier d'importation des guillemets aux noms des variables ou de variables référencées.

Exemple

- Nom de variable avec caractère spécial :
`<name>Siemens.Simatic.Hmi.Utah.Tag.HmiTag:41000_Options_Time_Date |DB_SFX0908_HMI1.Actual_Date_Time.Hour</name>`
- Nom de variable avec caractère spécial, entre guillemets :
`<name>"Siemens.Simatic.Hmi.Utah.Tag.HmiTag:41000_Options_Time_Date |DB_SFX0908_HMI1.Actual_Date_Time.Hour"</name>`

8.3.4.3 Exporter des variables individuelles d'une table de variables IHM

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Utilisation

Les types d'objet de modèle d'objet suivants peuvent exister sous la forme d'éléments subordonnés d'une variable HMI et sont pris en compte à l'exportation :

MultilingualText	Pour commentaire, TagValue, DisplayName
TagArrayMemberTag	Pour éléments de tableau IHM
TagStructureMember	Pour éléments de structure IHM
Event	Pour événements configurés
MultiplexEntry	Pour les entrées multiplex configurées de variables

Code du programme

Pour exporter une seule variable d'une table de variables IHM vers un fichier XML, modifiez le code de programme suivant :

```
//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);
    }
}
```

8.3.4.4 Importer des variables individuelles d'une table de variables IHM

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Utilisation

Les types d'objet de modèle d'objet suivants peuvent exister sous la forme d'éléments subordonnés d'une variable HMI et être pris en compte à l'importation :

MultilingualText	Pour commentaire, TagValue, DisplayName
TagArrayMemberTag	Pour éléments de tableau IHM
TagStructureMember	Pour éléments de structure IHM
Event	Pour événements configurés
MultiplexEntry	Pour les entrées multiplex configurées de variables

Code du programme

Pour importer une variable IHM dans une table de variables IHM depuis un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

8.3.4.5 Particularités de l'importation/exportation de variables IHM

Introduction

L'exportation/importation des variables IHM suivantes présentent des particularités :

- Variables IHM externes avec liaison intégrée
- Variables IHM avec le type de données "UDT"

Codes de programme similaires

Le code de programme pour les variables IHM susmentionnées est presque identique aux codes de programme suivants :

- Code du programme : Exportation de variables IHM (Page 396)
- Code du programme : Importation de variables IHM (Page 397)

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Particularités de l'importation/exportation d'une variable IHM externe avec liaison intégrée

Lors de l'exportation d'une variable IHM externe avec liaison IHM intégrée, seule la liaison des variables IHM à la variable API est enregistrée dans le fichier d'exportation, à la place des données de variables API.

Avant l'importation, assurez-vous que l'API, les variables API correspondantes et la liaison intégrée à l'API correspondant sont présents dans le projet. Si tel n'est pas le cas, il faut créer ces éléments avant de lancer l'importation. Lors de l'importation consécutive de la variable IHM externe, la liaison à la variable API est réactivée.

Les noms des variables IHM externes au-delà de toutes les tables de variables d'un projet doivent être univoques. Si vous n'indiquez pas la table de variables correspondant à la variable IHM lors de l'importation, l'importation est annulée.

Pour importer une variable IHM externe avec liaison intégrée, utilisez la structure XML suivante :

```
<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>
```

Particularités de l'importation/exportation d'une variable IHM du type de données "UDT"

Lors de l'exportation d'une variable IHM du type de données "UDT", le raccourci vers le type de données est exporté. Pour l'importation, seuls les types de données versionnés sont pris en charge.

Les types de données doivent être enregistrés dans la bibliothèque de projet. Les types de données de la bibliothèque globale ne sont pas pris en charge.

Les règles suivantes doivent être respectées pour l'importation :

- Les types de données référencés doivent figurer dans la bibliothèque de projet. L'importation est annulée si le type de données ne figurent pas dans la bibliothèque de projet.
- Le type de données référencé doit être versionné. L'attribution de versions est prise en charge à partir de TIA Portal V13 SP1. Si le type de données n'est pas versionné, une exception est déclenchée.

Remarque

Le premier type de données trouvé est utilisé pour la résolution de la référence lors de l'importation.

Tenez compte ici des points suivants : Le répertoire racine de la bibliothèque est d'abord parcouru, puis les sous-dossiers.

Pour importer une variable IHM du type de données "UDT", utilisez la structure XML suivante :

```
<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>
```

8.3.5 Scripts VB

8.3.5.1 Exporter des scripts VB

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal. Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert. Voir Ouvrir un projet (Page 99)

Utilisation

Tous les dossiers personnalisés de niveau inférieur sont pris en compte au cours de l'exportation. Pour chaque script VB exporté est créé un fichier XML spécifique.

Code du programme : Exporter un script VB

Pour exporter un script VB sélectionné d'un appareil IHM vers un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

8.3.5.2 Exporter des scripts VB à partir d'un dossier

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Pour chaque script VB exporté est créé un fichier XML spécifique.

Code du programme : exporter un script VB d'un dossier personnalisé

Pour exporter un script VB d'un dossier personnalisé vers un fichier XML, modifiez le code de programme suivant :

```
//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);
    }
}
```

Code du programme : Exporter tous les scripts VB à partir d'un dossier système

Pour exporter tous les scripts VB du dossier système, modifiez le code de programme suivant :

```
//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);
    }
}
```

8.3.5.3 Importer des scripts VB**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les importations groupées sont prises en charge : Sinon, vous pouvez aussi utiliser un code de programme avec une boucle Foreach (Exporter des scripts VB (Page 399)).

Code du programme

Pour importer un script VB dans un appareil IHM depuis un fichier XML, modifiez le code de programme suivant :

```
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);
    }
}
```

8.3.6 Listes de textes

8.3.6.1 Exporter des listes de textes à partir d'un appareil IHM

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'exportation de listes de textes et de graphiques inclut toutes les entrées des listes. Les listes de textes et de graphiques peuvent être exportées séparément.

Les listes de textes d'un appareil IHM sont exportées. Pour chaque liste de textes exportée, un fichier XML spécifique est créé.

Code du programme

Pour exporter des listes de textes d'un appareil IHM, modifiez le code de programme suivant :

```
//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);
    }
}
```

8.3.6.2 Importer une liste de texte dans un appareil IHM

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'interface API prend en charge l'importation d'une liste de textes dans un appareil IHM depuis un fichier XML.

Code du programme

Pour importer une liste de textes dans un appareil IHM depuis un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

8.3.6.3 Formats XML avancés pour l'exportation/importation de listes de textes

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- Exportation standard de listes de textes
Voir Exporter des listes de textes à partir d'un pupitre opérateur (Page 402)
- Importation standard de listes de textes
Voir Importer des listes de textes dans un pupitre opérateur (Page 403)

Utilisation

Une liste de textes peut aussi contenir des textes formatés. Cela concerne pour l'essentiel les formatages suivants :

- Formatage de texte
- Références aux autres objets dans le texte

Les formatages textuels purs dans une liste de textes à exporter conduisent à un format d'exportation XML étendu. Les références aux objets sont exprimés sous la forme d'Open Links. De même que les listes de textes à importer avec des textes formatés.

Les formats d'exportation XML étendus peuvent aussi nettement se complexifier. A titre d'exemple, d'autres liens que le seul nom de l'objet peuvent parfois exister dans la liste de textes, p. ex. via un Open Link vers une variable API d'un autre appareil. Si tel est le cas, toutes les informations doivent être codées en une chaîne de caractères pour supprimer l'Open Link.

```

<?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_li
                </reference>
                <subreference TargetID="@OpenLink">
                  <name>Siemens.Simatic.Hmi.Utah.Tag.HmiTag:t1</name>
                </subreference>
                <domaindata>
                  <format length="9" />
                </domaindata>
              </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_li
                </reference>
                <subreference TargetID="@OpenLink">
                  <name>Siemens.Simatic.Hmi.Utah.Tag.HmiTag:t1</name>
                </subreference>
                <domaindata>
                  <format length="9" />
                </domaindata>
              </fieldinfo>
            </fieldinfos>
          </Text>
        </AttributeList>
      </MultilingualTextItem>
    </ObjectList>
  </MultilingualText>

```

8.3.7 Listes de graphiques

8.3.7.1 Exporter les listes de graphiques

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'exportation de listes de textes et de graphiques inclut toutes les entrées des listes. Les listes de textes et de graphiques peuvent être exportées séparément.

Un fichier XML est créé par liste de graphiques. Les objets graphiques globaux contenus dans les listes de graphiques sont exportés sous la forme d'Open Links.

Code du programme

Pour exporter des listes de graphiques d'un pupitre opérateur, modifiez le code de programme suivant :

```
//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);
    }
}
```

8.3.7.2 Importer les listes de graphiques

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

L'interface API prend en charge l'importation d'une liste de graphiques dans un appareil IHM depuis un fichier XML.

Tous les objets graphiques référencés de la liste de graphiques sont inclus dans l'importation. Les références aux graphiques globaux ne sont pas incluses. Si les graphiques globaux référencés existent dans le projet cible, les références aux graphiques globaux sont rétablies lors de l'importation.

Code du programme

Pour importer une liste de graphiques dans un appareil IHM depuis un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

8.3.8 Connexions

8.3.8.1 Exporter des connexions

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Utilisation

L'interface API prend en charge l'exportation de toutes les liaisons d'un appareil IHM vers un fichier XML.

Remarque

Exporter des connexions intégrées

L'exportation de connexions intégrées n'est pas prise en charge.

Pour chaque connexion exportée, un fichier XML spécifique est créé.

8.3 Importation/exportation de données d'un appareil IHM

Code du programme

Pour exporter toutes les connexions d'un appareil IHM vers un fichier XML, modifiez le code de programme suivant :

```
//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));
        connection.Export(info, ExportOptions.WithDefaults);
    }
}
```

8.3.8.2 Importation de connexions

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Utilisation

L'interface API prend en charge l'importation de toutes les liaisons d'un appareil IHM dans un appareil IHM depuis un fichier XML. Si vous souhaitez importer plusieurs liaisons de communication, importez à chaque fois le fichier XML pour la connexion correspondante.

Remarque

Si vous importez une liaison dans un projet dans lequel une liaison intégrée est déjà configurée, cette liaison n'est pas écrasée. L'importation est annulée et une Exception est déclenchée.

Code du programme

Pour importer une seule liaison d'un appareil IHM dans un appareil IHM depuis un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

8.3.9 Vues

8.3.9.1 Vue d'ensemble des objets graphiques pouvant être exportés

Utilisation

Vous pouvez exporter et importer les vues suivantes par le biais d'API TIA Portal Openness :

Tableau 8-4 Vues prises en charge

Objet	Exportation/importation possible
Vue	Oui
Vue globale	Oui
Modèle de vue	Oui
Fenêtre permanente	Oui
Vue contextuelle	Oui
Vue Slide-in	Oui

Vous pouvez exporter ou importer les objets de vue suivants par le biais d'API TIA Portal Openness :

Tableau 8-5 Objets graphiques pris en charge

Zone	Type d'objet	Exportation/importation possible
Objets simples	Ligne	Oui
	Ligne polygonale	Oui
	Polygone	Oui
	Ellipse	Oui
	Segment d'ellipse	–
	Segment de cercle	–
	Arc d'ellipse	–
	Arc de cercle	–
	Cercle	Oui
	Rectangle	Oui
	Connecteur	–
	Champ de texte	Oui
	Affichage graphique	Oui
	Tuyau	–
	Double raccord en T	–
	Raccord en T	–
Coude	–	

Zone	Type d'objet	Exportation/importation possible
Eléments	Champ d'E/S	Oui
	Champ d'E/S graphique	Oui
	Champ de texte éditable	–
	Champ de liste	–
	Zone de liste déroulante	–
	Bouton	Oui
	Bouton rond	–
	Bouton-poussoir lumineux	Oui
	Commutateur	Oui
	Champ d'E/S symbolique	Oui
	Champ date/heure	Oui
	Bargraphe	Oui
	Bibliothèque d'icônes	Oui
	Curseur	Oui
	Barre de défilement	–
	Case à cocher	–
	Bouton d'option	–
	Instrument à aiguille	Oui
	Horloge	Oui
	Vue de l'espace mémoire	–
	Touches de fonction (touches programmables)	Oui
	Groupes	Oui
	Instances de bloc d'affichage	Oui

Zone	Type d'objet	Exportation/importation possible
Eléments de commande	Fenêtre de vues	–
	Vue des utilisateurs	Oui
	Travail d'impression/Diagnostic de script	–
	Affichage caméra	–
	Affichage PDF	–
	Vue de recette	–
	Vue des alarmes	–
	Indicateur d'alarme	–
	Fenêtre d'alarmes	–
	Vue de courbes f(x)	–
	Vue de courbes f(t)	–
	Vue tabellaire	–
	Table des valeurs	–
	Navigateur HTML	–
	Media Player	–
	Diagnostic de voie	–
	WLAN - Réception	–
	Zone - Nom	–
	Zone - Signal	–
	Nom de la plage d'action	–
	Nom de la plage d'action (RFID)	–
	Signal de la plage d'action	–
	Etat de chargement	–
	Molette	–
	Indicateur d'aide	–
	Vue Sm@rtClient	–
	Visualisation/forçage	–
	Vue de l'espace mémoire	–
	Affichage de section de programme NC	–
	Vue de diagnostic système	–
Fenêtre de diagnostic système	–	

Voir aussi

Notions élémentaires sur l'importation/exportation (Page 371)

8.3.9.2 Exporter toutes les vues d'un appareil IHM

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Un autre code de programme est nécessaire pour exporter toutes les vues agrégées de tous les dossiers personnalisés d'un appareil IHM.

Code du programme : Exporter toutes les vues d'un appareil

Pour exporter les vues d'un dossier de vues personnalisé d'un appareil IHM et le dossier de vues système, modifiez le code de programme suivant :

```
private static void ExportScreensOfDevice(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);
}
```

Code de programme : Exporter toutes les vues d'un dossier personnalisé

Pour exporter les vues d'un dossier de vues personnalisé d'un appareil IHM et le dossier de vues système, modifiez le code de programme suivant :

```
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);
    }
}
```

Code du programme : exporter toutes les vues d'un appareil quel que soit l'utilisateur

Pour exporter toutes les vues, modifiez le code de programme suivant :

```
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);
    }
}

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);
    }
}
```

8.3.9.3 Exporter une vue à partir d'un dossier de vues

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes d'une vue sont exportées :

Vue	Données
Attributs	ActiveLayer, BackColor, Height, Width, Name, Number, HelpText
Ouvrir des liens	Template
Compositions	<ul style="list-style-type: none"> • Layers • Animations Toutes les animations basées sur Runtime Advanced configurées sont exportées. • Events Tous les événement basés sur Runtime Advanced configurés sont exportés. • Softkeys Toutes les touches programmables configurées sont exportées.

Pour chaque couche, les données suivantes sont exportées :

Remarque

le nom de la couche dans TIA Portal est un texte vide par défaut.

Si vous ne modifiez pas le nom de la couche dans TIA Portal, le nom de la couche exportée est vide. Dans ce cas, le nom de la couche affiché dans TIA Portal dépend de la langue de l'interface utilisateur.

Si vous modifiez le nom de la couche dans TIA Portal, le nom modifié sera affiché dans toutes les langues correspondantes.

Couche	Données
Attributs	Name, Index, VisibleES
Compositions	ScreenItems (avec éléments graphique)

Les éléments suivants ne sont pas inclus dans l'exportation :

- Attributs spécifiques SCADA
- Couches qui ne contiennent pas d'éléments graphiques et dont les attributs ne se distinguent pas des valeurs par défaut.

Code du programme

Pour exporter une seule vue à partir du dossier utilisateur ou du dossier système d'un appareil IHM, modifiez le code de programme suivant :

```
//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);
    }
}
```

8.3.9.4 Importer des vues dans un appareil IHM

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les vues ne peuvent être importées que dans un type donné d'appareil IHM. L'appareil IHM et l'appareil à partir duquel les vues ont été exportées sont du même type d'appareil.

Les données suivantes d'une vue sont exportées :

Vue	Données
Attributs	ActiveLayer, BackColor, Height, Width, Name, Number, HelpText
Ouvrir des liens	Templates
Compositions	<ul style="list-style-type: none">• Layers• Animations Toutes les animations configurables pour des vues sont importées.• Events Toutes les animations configurables pour des événements sont importées.• Softkeys Toutes les animations configurables pour des touches programmables sont importées.

Pour chaque couche, les données suivantes sont importées :

Remarque

Si vous avez indiqué un texte vide pour le nom de la couche avant l'importation, le nom de la couche affiché dans TIA Portal dépend de la langue de l'interface utilisateur après l'importation.

Si vous avez attribué un nom à la couche, le nom indiqué est affiché dans toutes les langues correspondantes après l'importation.

Couche	Données
Attributs	Name, Index
Compositions	ScreenItems

Restrictions

- Lorsque la largeur et la hauteur d'une vue ne correspondent pas aux dimensions de l'appareil, le processus d'importation est interrompu et une Exception est déclenchée. L'ajustement des éléments graphiques compris n'est pas pris en charge. C'est pourquoi, certains éléments graphiques peuvent se trouver en-dehors des limites de la vue. Si tel est le cas, un avertissement du compilateur est émis.
- Le numéro de vue doit être univoque pour toutes les vues de l'appareil. L'importation d'une vue est annulée si une vue avec un numéro de vue qui a déjà été créé dans l'appareil, est trouvé. Si vous n'avez pas encore attribué de numéro à la vue, un numéro de vue univoque est affecté à la vue pendant le processus d'importation.
- L'ordre des éléments graphiques au sein de l'ordre Z doit être univoque et sans lacunes pour chaque couche dans la vue. C'est pourquoi une vérification de la cohérence, qui répare l'ordre si nécessaire, est effectuée après l'importation de la vue. Ce processus peut entraîner la modification d'"Indices de tabulation" pour certains éléments graphiques. Vous pouvez modifier manuellement l'ordre Z des éléments graphiques dans le fichier XML. L'élément graphique au premier emplacement se trouve tout à la fin de l'ordre Z.

Remarque

Vous pouvez modifier les valeurs de largeur et hauteur d'un élément graphique dans le fichier XML si l'attribut "Adapter la taille au contenu" est activée pour l'élément graphique.

Remarque

L'importation de types de vue de la bibliothèque n'est pas prise en charge

À partir de WinCC V12 SP1, vous pouvez créer une vue en tant que type dans la bibliothèque. Les instances du type de vue utilisées dans le projet peuvent être éditées avec l'application TIA Portal Openness comme d'autres vues. Si vous exportez des vues, les instances des types de vue sont exportées sans les informations de type.

Si vous réimportez ces vues dans le projet, les instances du type de vue sont écrasées et l'instance est remplacée par le type de vue.

Code du programme : Importer des vues dans un appareil IHM

Pour importer des vues avec la boucle For each dans un appareil IHM, modifiez le code de programme suivant :

```
//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);
    }
}
```

Code du programme : Importer dans un dossier utilisateurs nouvellement créé

Pour importer une vue dans un dossier utilisateur venant d'être créé d'un appareil IHM, modifiez le code de programme suivant :

```
//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);
}
```

8.3.9.5 Exporter une fenêtre permanente

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes de la fenêtre permanente sont exportées :

Fenêtre permanente	Données
Attributs	ActiveLayer, BackColor, Height, Width, Name
Compositions	Layers

Pour chaque couche, les données suivantes sont exportées :

Couche	Données
Attributs	Name, Index
Compositions	ScreenItems (avec éléments graphiques)

Code du programme

Pour exporter une fenêtre permanente d'un appareil IHM vers un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

8.3.9.6 Importer une fenêtre permanente

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes de la fenêtre permanente sont importées :

Fenêtre permanente	Données
Attributs	ActiveLayer, BackColor, Height, Width, Name, Visible, Number
Compositions	Layers

8.3 Importation/exportation de données d'un appareil IHM

Pour chaque couche, les données suivantes sont importées :

Couche	Données
Attributs	Name, Index
Compositions	ScreenItems (avec éléments graphiques)

Lorsque la largeur et la hauteur d'une vue ne correspondent pas aux dimensions de l'appareil, le processus d'importation est interrompu et une Exception est déclenchée. L'ajustement des éléments d'appareil compris (éléments graphiques) n'est pas pris en charge. C'est pourquoi, certains éléments d'appareil peuvent se trouver en-dehors des limites de la vue. Si tel est le cas, un avertissement du compilateur est émis.

L'ordre des éléments d'appareil dans la fenêtre permanente doit être univoque et ne présenter aucune lacune. C'est pourquoi une vérification de la cohérence, qui répare l'ordre si nécessaire, est effectuée après l'importation de la fenêtre permanente. Ce processus peut entraîner la modification d'"Indices de tabulation" pour certains éléments d'appareil.

Code du programme

Pour importer une fenêtre permanente dans un appareil IHM depuis un fichier XML, modifiez le code de programme suivant :

```
//Imports a permanent area
private static void ImportScreenOverview(HmiTarget hmiTarget)
{
    FileInfo info = new FileInfo(@"D:\Samples\Screens\ExportedOverview.xml");
    hmiTarget.ImportScreenOverview(info, ImportOptions.Override);
}
```

8.3.9.7 Exporter tous les modèles de vue d'un appareil IHM

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal. Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert. Voir Ouvrir un projet (Page 99)

Utilisation

Un fichier XML est créé par modèle de vue.

Les exportations groupées n'étant pas prises en charge, vous devez énumérer tous les modèles de vue et les exporter séparément. Ce faisant, veillez à ce que les noms utilisés pour les modèles de vue correspondent aux conventions de dénomination de fichiers de votre système de fichiers.

Code de programme : exporter tous les modèles de vue d'un appareil

Pour exporter tous les modèles de vue d'un certain dossier, modifiez le code de programme suivant :

```
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);
}
```

Code de programme : exporter tous les modèles de vue d'un dossier défini

Pour exporter tous les modèles de vue, modifiez le code de programme suivant :

```
//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);
    }
}
```

8.3.9.8 Exporter des modèles de vue à partir d'un dossier**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes du modèle de vue sont exportées :

Modèles de vue	Données
Attributs	ActiveLayer, BackColor, Height, Width, Name
Compositions	<ul style="list-style-type: none"> • Layers • Animations Toutes les animations configurées sont exportées. Les animations SCADA ne sont pas exportées. • Softkeys Toutes les touches programmables configurées sont exportées.

Pour chaque couche, les données suivantes sont exportées :

Couche	Données
Attributs	Name, Index
Compositions	ScreenItems (avec éléments graphiques)

Code du programme : exporter un modèle de vue d'un dossier personnalisé

Pour exporter un seul modèle de vue à partir du dossier système ou d'un dossier personnalisé, modifiez le code de programme suivant :

```
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);
}
```

Code du programme : exporter tous les modèles de vue d'un dossier personnalisé

Pour exporter tous les modèles de vue d'un certain dossier, modifiez le code de programme suivant :

```
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);
    }
}
```

Code de programme : exporter tous les modèles de vue d'un dossier défini

Pour exporter tous les modèles de vue, modifiez le code de programme suivant :

```
//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 folders.Folders)
    {
        ExportScreenTemplates(Path.Combine(templatePath, subfolder.Name), subfolder);
    }
}
```

8.3.9.9 Importer des modèles de vue**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes d'un modèle de vue sont importées :

Modèle de vue	Données
Attributs	ActiveLayer, BackColor, Height, Width, Name, SetTabOrderInFront
Compositions	<ul style="list-style-type: none"> • Layers • Animations Toutes les animations configurables pour des vues sont importées. • Softkeys Toutes les animations configurables pour des touches programmables sont importées.

Pour chaque couche, les données suivantes sont importées :

Couche	Données
Attributs	Name, Index
Compositions	ScreenItems (avec éléments graphiques)

Lorsque la largeur et la hauteur d'un modèle de vue ne correspondent pas aux dimensions de l'appareil, le processus d'importation est interrompu et une Exception est déclenchée. L'ajustement des éléments graphiques compris n'est pas pris en charge. C'est pourquoi, certains éléments graphiques peuvent se trouver en-dehors des limites de la vue. Si tel est le cas, un avertissement du compilateur est émis.

L'ordre des éléments d'appareil dans le modèle de vue doit être univoque et ne présenter aucune lacune. C'est pourquoi une vérification de la cohérence, qui répare l'ordre si nécessaire, est effectuée après l'importation du modèle de vue. Ce processus peut entraîner la modification d'"Indices de tabulation" pour certains éléments graphiques.

Code du programme : Importation générale

Pour importer tous les modèles de vues avec la boucle For each dans un appareil IHM, modifiez le code de programme suivant :

```
//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);
    }
}
```


Code du programme : Importer dans un dossier utilisateurs nouvellement créé

Pour importer un modèle de vue dans un dossier utilisateur venant d'être créé d'un appareil IHM, modifiez le code de programme suivant :

```
//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);
}
```

8.3.9.10 Exportation d'une vue contextuelle

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes de la vue contextuelle sont exportées :

Modèles de vue	Données
Attributs	ActiveLayer, BackColor, GridColor, Height, Name, ScrollbarBackgroundColor, ScrollbarForegroundColor, Width
Compositions	<ul style="list-style-type: none"> • Layers • Events Tous les événements configurés sont exportés.

Pour chaque couche, les données suivantes sont exportées :

Couche	Données
Attributs	Name, Index, VisibleES
Compositions	ScreenItems Tous les objets graphiques exportables sont exportés.

Code du programme : Exportation d'une vue contextuelle à partir d'un dossier

Pour exporter une seule vue contextuelle à partir du dossier système ou d'un dossier personnalisé, modifiez le code de programme suivant :

```
//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);
}
```

8.3.9.11 Importation d'une vue contextuelle

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes de la vue contextuelle sont importées :

Modèles de vue	Données
Attributs	ActiveLayer, BackColor, GridColor, Height, Name, ScrollbarBackgroundColor, ScrollbarForegroundColor, Width
Compositions	<ul style="list-style-type: none">• Layers• Events Tous les événements configurés sont exportés.

L'existence des attributs suivants est obligatoire pour l'importation :

- Name
- Height
- Width

Pour chaque couche, les données suivantes sont importées :

Couche	Données
Attributs	Name, Index, VisibleES
Compositions	ScreenItems Tous les objets graphiques importables sont importés.

Restrictions

Lorsque l'appareil ne prend pas en charge les vues contextuelles, le processus de copie est annulé et une exception est déclenchée.

Lorsque la largeur et la hauteur d'une vue contextuelle ne correspondent pas aux dimensions de l'appareil, le processus d'importation est interrompu et une Exception est déclenchée.

- Hauteur minimale = 1 pixel
- Largeur minimale = 1 pixel
- Hauteur maximale = six fois la hauteur de l'écran de l'appareil
- Largeur maximale = deux fois la largeur de l'écran de l'appareil
- Pour les appareils avec la version de Runtime V13 SP1, la hauteur et la largeur maximales correspondent à la hauteur et à la largeur de l'écran de l'appareil.

Code du programme : Importation d'une vue contextuelle dans un dossier

Pour importer une vue contextuelle dans un dossier système de vue contextuelle ou dans un dossier personnalisé, modifiez le code de programme suivant :

```
//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);
}
```

8.3.9.12 Exportation d'une vue encastrable

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données et valeurs suivantes de la vue Slide-in sont exportées :

Modèles de vue	Données	
Attributs	Activate	false
	ActiveLayer	0
	BackColor	(182 ; 182 ; 182)
	GridColor	(0, 0, 0)
	Dimension	427 L'attribut "Dimension" indique soit la largeur, soit la hauteur de la vue Slide-in, en fonction du type de vue encastrable.
	LineColor1	(223 ; 223 ; 223)
	LineColor2	(32 ; 32 ; 32)
	OperatableAreaColor	(128 ; 128 ; 128)
	SlideinType	En haut, en bas, à gauche, à droite Les vues Slide-in n'ont pas de nom mais un SlideinType.
Visibility	FadeOut	
Compositions	Layers	

Remarque

Les vues Slide-in n'ont pas de nom mais un SlideinType.

Pour chaque couche, les données suivantes sont exportées :

Couche	Données	
Attributs	Name,	
	Index	
	VisibleES	
Compositions	ScreenItems	Tous les objets graphiques exportables sont exportés.

Code du programme : Exportation d'une vue Slide-in

Pour exporter une seule vue Slide-in du dossier système, modifiez le code de programme suivant :

```
//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);
}
```

8.3.9.13 Importation d'une vue encastrable

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données et valeurs suivantes d'une vue Slide-in sont importées :

Modèles de vue	Données
Attributs	Activate = false ActiveLayer = 0 Authorization BackColor = (182; 182; 182) Dimension = 427 L'attribut "Dimension" indique soit la largeur soit la hauteur de la vue Slide-in, en fonction de celui des deux attributs qui peut être modifié pour le type de la vue Slide-in. GridColor = (0; 0; 0) LineColor1 = (223; 223; 223) LineColor2 = (32; 32; 32) OperateableAreaColor = (128; 128; 128) SlideinType = Top, Bottom, Left, Right Visibility = FadeOut
Compositions	Layers

L'existence de l'attribut suivant est obligatoire pour l'importation :

- SlideinType

Pour chaque couche, les données suivantes sont importées :

Couche	Données
Attributs	Name, Index, VisibleES
Compositions	ScreenItems Tous les objets graphiques importables sont importés.

Restrictions

- Lorsque l'appareil ne prend pas en charge les vues Slide-in, l'importation est annulée et une exception est déclenchée.
- Si une vue Slide-in est référencée par un autre élément, celle-ci doit être référencée par openlink et non par SlideinType, par ex. dans la fonction système "ShowSlideinScreen"). Le tableau suivant représente l'attribut "SlideinType" avec l'openlink correspondant :

SlideinType	Nom de l'openlink
Top	GraphX_Slidein_Top
Right	GraphX_Slidein_Right
Bottom	GraphX_Slidein_Bottom
Left	GraphX_Slidein_Left

Code du programme : Importation d'une vue Slide-in dans un dossier

Pour importer une vue Slide-in dans un dossier système de vue Slide-in, modifiez le code de programme suivant :

```
//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);
}
```

8.3.9.14 Exportation d'une vue avec masque de saisie

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes d'une instance de bloc d'affichage sont exportées dans une vue :

Vue	Données
Attributs	Left, Top, Width, Height, ObjectName, Resizing, TabIndex, FaceplateTypeName
Attributs d'interfaces	Tous les attributs d'interface configurés d'une instance de bloc d'affichage sont exportés en tant qu'éléments graphiques exportables.
Compositions	<ul style="list-style-type: none"> • Animations Toutes les images animées sont exportées. Les animations de variables sont basées sur les attributs de l'interface. • Événements Tous les événements configurés sont exportés.

Tenez compte des consignes suivantes pour les attributs exportés d'instances de blocs d'affichage :

- Resizing
L'attribut "Resizing" est exporté dans tous les cas, quelles que soient les options d'exportation.
- FaceplateTypeName
L'attribut "FaceplateTypeName" identifie le type correspondant et la version du bloc d'affichage, par ex. "Faceplate_1 V 0.0.2".

Remarque

Type de bloc d'affichage dans un dossier de bibliothèque

Si un type de bloc d'affichage se trouve dans un dossier de bibliothèque, vous avez besoin du chemin et du nom complets pour identifier le type de bloc d'affichage. Le mot-clé "@ \$@" est utilisé pour séparer les dossiers et/ou les noms des types de bloc d'affichage, par ex. "Folder_1@\$@SubFolder_1@\$@Faceplate_1 V 0.0.2".

Les données suivantes d'éléments graphiques dans une instance de bloc d'affichage sont exclues de l'exportation :

Élément graphique	Attribut
Champ d'E/S	Flashing on limit violation
Champ d'E/S graphique	Fit embedded graphic object to screen size

Code du programme

Pour exporter une seule vue avec une instance de bloc d'affichage, modifiez le code de programme suivant :

```
//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);
    }
}
```

8.3.9.15 Importation d'une vue avec masque de saisie

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Les données suivantes d'une instance de bloc d'affichage sont importées sur une vue :

Vue	Données
Attributs	Left, Top, Width, Height, ObjectName, Resizing, TabIndex, FaceplateTypeName
Attributs d'interfaces	Tous les attributs d'interface configurés d'une instance de bloc d'affichage sont importés en tant qu'éléments graphiques importables.
Compositions	<ul style="list-style-type: none">• Animations Toutes les images animées sont importées. Les animations de variables sont basées sur les attributs de l'interface.• Événements Tous les événement configurés sont importés.

L'existence des attributs suivants est obligatoire pour l'importation :

- ObjectName
- FaceplateTypeName

Les données suivantes d'éléments graphiques dans une instance de bloc d'affichage sont exclues de l'exportation et de l'importation :

Élément graphique	Attribut
Champ d'E/S	Flashing on limit violation
Champ d'E/S graphique	Fit embedded graphic object to screen size

Restrictions

- Bloc d'affichage, événement ou attribut de l'interface inconnu
Si un faceplate type name, un event name ou un interface attribute name n'existant pas dans le projet est indiqué dans le fichier d'importation, l'importation est annulée et une exception est déclenchée.
- Comportement de mise à l'échelle d'une instance de bloc d'affichage
L'attribut "Resizing" est importé dans tous les cas, quelles que soient les options d'exportation.
Exemples :
Si "Resizing" est réglé sur "KeepRatio", l'attribut "Height" est utilisé pour calculer la valeur d'attribut "Width".
 - La taille d'un type de bloc d'affichage est de 100 x 100 pixels. Si une instance de bloc d'affichage ayant une taille de 300 x 100 pixels est importée et que la valeur "FixedSize" est paramétrée sur l'attribut "Resizing", l'importation est réussie et la taille du bloc d'affichage est réglée sur 100 x 100 pixels.
 - La taille d'un type de bloc d'affichage est de 100 x 50 pixels. Si une instance de bloc d'affichage ayant une taille de 100 x 100 pixels est importée et que la valeur "KeepRatio" est paramétrée sur l'attribut "Resizing", l'importation est réussie et la taille du bloc d'affichage est réglée sur 200 x 100 pixels.

Remarque

Comportement de mise à l'échelle des instances de bloc d'affichage importées

Les valeurs de "Resizing" et les valeurs des attributs d'interface peuvent influencer sur la taille des instances de bloc d'affichage importées et même sur la taille des éléments graphiques qui y sont incorporés.

Pour éviter toute modification intempestive de la mise en forme d'une instance de bloc d'affichage, importez-la dans sa taille initiale et sans les valeurs d'attribut "Width" et "Height".

- Valeurs d'attribut d'interfaces différentes
 - Si vous modifiez les attributs pour l'importation, la dernière valeur d'attribut d'interface utilisée est importée.
 - En cas d'interdépendance des attributs, il se peut que d'autres attributs soient modifiés au cours de l'importation.
Exemple : Un bloc d'affichage contient un champ d'E/S. L'attribut "Mode de fonctionnement" est connecté à un attribut d'interface. Si vous réglez d'abord le mode de fonctionnement sur "Sortie" puis l'attribut "Entrée cachée" sur Vrai, la valeur de l'"Entrée cachée" n'est pas utilisée après l'importation. L'attribut "Entrée cachée" a été mis en lecture seule lors de la première modification et la valeur ne peut pas être utilisée.
 - Si une valeur d'attribut ne respecte pas les restrictions imposées par WinCC, la valeur du type de bloc d'affichage s'affiche.
Exemple : La plage d'affichage d'un instrument à aiguille est paramétrée sur 10 - 80. Les attributs "Maximum" et "Minimum" sont configurés comme attributs d'interfaces. Si vous paramétrez une valeur minimale qui dépasse la valeur maximale, par ex. 100, la valeur du type de bloc d'affichage pour "Minimum" s'affiche après l'importation.

- Si un attribut d'interface est associé à plusieurs attributs d'éléments graphiques au sein du type de bloc d'affichage, l'attribut d'interface sur l'instance de bloc d'affichage affiche alors la valeur d'attribut utilisée du premier élément graphique associé.
Exemple : Un bloc d'affichage contient deux instruments à aiguille avec des valeurs maximales différentes. Les valeurs minimales de ces deux instruments sont reliées à un seul attribut d'interface.
Si vous paramétrez une seule valeur minimale utilisable pour les deux instruments à aiguille, les deux valeurs sont paramétrées.
Si vous paramétrez une valeur utilisable uniquement pour le deuxième instrument à aiguille, la valeur est paramétrée uniquement pour celui-ci. En revanche, la valeur du premier instrument à aiguille s'affiche comme attribut d'interface.

Code du programme : Importer des images avec une instance de bloc d'affichage

Pour importer une vue avec une instance de bloc d'affichage, modifiez le code de programme suivant :

```
//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);
}
```

8.4 Importation/exportation de données d'un appareil API

8.4.1 Blocs

8.4.1.1 Structure XML de la section interface du bloc

Principe de base

Les données du fichier d'exportation issues de l'importation/exportation sont organisées au moyen d'une structure de base. Chaque fichier d'importation doit répondre aux conditions structurelles de base.

Le fichier d'exportation contient toutes les variables et constantes traitées de la section d'interface d'un bloc exporté. Tous les attributs avec `"ReadOnly = "TRUE"` et `"Informative = "TRUE"` sont exclus.

Quand les informations sont redondantes, elles doivent être identiques dans le fichier XML d'importation et dans le fichier de projet. Autrement, l'importation lance une exception récupérable.

Les données de projet contiennent plus de données que le fichier XML d'importation, par ex. un type externe peut avoir des membres supplémentaires.

Seules les valeurs accessibles en écriture peuvent être importées via TIA Portal Openness XML.

En fonction des paramètres d'exportation réglés dans TIA Portal Openness, le fichier d'exportation contient un jeu déterminé d'attributs et d'éléments. Le fichier XML exporté avec des versions ultérieures du produit n'est pas compatible avec la procédure d'importation dans des versions antérieures de TIA Portal.

Structure de base

La section d'interface d'un bloc exporté est contenue dans l'élément `<Interface>` dans la SimaticML d'un bloc. L'objet racine est l'élément `<Sections>` qui représente la section d'interface d'un bloc exporté. Dans la description ci-dessous, les éléments figurent dans l'ordre requis pour le fichier de saisie.

```
<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>
```

- **Section**
La section représente un paramètre particulier ou des données locales d'un bloc de programme.
- **Membre**
Le membre représente les variables ou constantes utilisées dans le bloc de programme. Selon le type de données d'une variable, les membres peuvent être imbriqués ou posséder d'autres sous-éléments structurels.
Par exemple, pour le type de données "ARRAY", l'élément structurel "Subelement Path" représente l'indice de composants d'un élément du tableau.
Seuls sont exportés les membres qui ont été édités par l'utilisateur.
- **AttributeList**
La `<AttributeList>` englobe tous les attributs définis d'un membre. Les attributs déterminés par le système ou affectés d'une valeur par défaut ne figurent pas dans la structure XML.
Les attributs de membre `<ReadOnly>` et `<Informative>` ne sont écrits dans le fichier d'exportation XML que s'ils ont la valeur VRAI.

8.4 Importation/exportation de données d'un appareil API

- StartValue
L'élément <StartValue> est écrit seulement quand la valeur par défaut de la variable ou de la constante est déterminée par l'utilisateur.

```
...  
<Member Name="Static_1" Datatype="Int">  
...  
  <StartValue>1</StartValue>  
</Member>  
...
```

- Commentaire
L'élément <Comment> est écrit quand il est déterminé par l'utilisateur. Les commentaires d'une variable ou d'une constante sont exportés comme textes multilingues :

```
...  
<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>  
...
```

Attributs

- Attributs principaux

Les attributs principaux sont écrits dans l'élément `<Member>` dans la structure XML.

```
...
<Member Name="Static_1" Datatype="User_data_type_1" Remanence="Retain">
...
</Member>
...
```

Le tableau ci-dessous indique les attributs principaux d'une variable ou d'une constante dans la section Interface du bloc.

Nom	Type de données	Valeur par défaut	Condition d'importation	Commentaire
Nom	STRING	-	Nécessaire	
Type de données	ENUM	-	Nécessaire	
Version	STRING	-	Optionnel	
Rémanence	ENUM	NonRetain	-	N'est écrit que s'il ne s'agit pas de u paramétrage par défaut
Accessibilité	ENUM	Public	-	Prédéfini par le système, ne peut être modifié par l'utilisateur
Informatif	BOOL	FALSE	-	

Les membres avec le memento "Informative" ne sont pas pris en considération lors de l'importation. Quand l'attribut est effacé ou mis à FALSE, une exception est lancée.

Remarque

Valeur de rémanence "Positionner dans IDB"

Quand une variable ou une constante a la valeur de rémanence "Positionner dans IDB", la rémanence déterminée dans le bloc de données d'instance doit être la même pour toutes les autres variables et constantes dont la rémanence est "SetInIDB".

Le premier membre importé avec l'attribut "Positionner dans IDB" détermine la rémanence attendue dans le bloc de données d'instance pour les variables et constantes suivantes dont la rémanence est "SetInIDB".

- Attributs de membre définis par le système

Les attributs de membre définis par le système sont listés dans l'élément `<AttributeList>`. Les attributs de membre définis par le système sont repérés par le memento `<Informative>` et ne sont pas pris en considération lors de l'importation.

8.4 Importation/exportation de données d'un appareil API

```

...
<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	Type	Valeur par défaut	SimaticML protégé en écriture (informatif)	Commentaire
At	String	""	FALSE	Des parts du membre alternent avec un autre membre dans cette structure
SetPoint	Bool	FALSE	FALSE	Le membre peut être synchronisé avec la mémoire de travail
UserReadOnly	Bool	FALSE	TRUE	L'utilisateur ne peut pas modifier les attributs de membre (y compris le nom)
UserDeletable	Bool	TRUE	TRUE	Le membre ne peut pas être supprimé dans l'éditeur
HmiAccessible	Bool	TRUE	FALSE	Pas d'accès HMI, pas d'élément de structure
HmiVisible	Bool	TRUE	FALSE	Filtre pour réduire le nombre de membres affichés d'abord
Offset	Int	-	TRUE	DB, FB, FC (temp). Pour les API classiques et les API Plus où la rémanence classique est réglée.

Name	Type	Valeur par défaut	SimaticML protégé en écriture (informatif)	Commentaire
PaddedSize	Int	-	TRUE	DB, FB, FC (temp). Pour les API classiques et les API Plus où la rémanence classique est réglée. Uniquement pour tableaux.
HiddenAssignment	Bool	FALSE	FALSE	Masquer l'affectation lors de l'appel quand elle correspond à PredefinedAssignment.
PredefinedAssignment	String	""	FALSE	Saisie pour le paramètre utilisé quand l'appel se trouve placé.
ReadOnlyAssignment	Bool	FALSE	FALSE	Lors de l'appel, l'utilisateur ne peut pas modifier l'affectation prédéfinie.
UserVisible	Bool	TRUE	TRUE	Ce membre ne s'affiche pas dans l'interface utilisateur.
HmiReadOnly	Bool	TRUE	TRUE	Ce membre est protégé en écriture pour HMI.
CodeReadOnly	Bool	FALSE	TRUE	-

- **Attributs définis par l'utilisateur**
 Les attributs définis par l'utilisateur sont repérés par le memento <ReadOnly>. Les membres avec ce memento ne sont pas pris en considération lors de l'importation. Quand le memento est effacé ou mis à FALSE, une exception est lancée.
 Les attributs définis par l'utilisateur non édités sont exclus de l'exportation.

Nom	Type	Valeur par défaut	SimaticML protégé en écriture (informatif)	Commentaire
CFC	IBlockAttribute	---	FALSE	Ceci est un chargement

Type de données "STRUCT"

Les composants du type de données "STRUCT" sont représentés comme membres imbriqués dans la structure XML d'un fichier d'importation/exportation :

```
<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>
```

Type de données "ARRAY", type de base

Les composants du type de données de base "ARRAY" sont représentés comme sous-éléments avec l'attribut "Path" dans la structure XML d'un fichier d'importation/exportation :

```
<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>
```

Type de données "ARRAY", d'un UDT

Les composants du type de données "ARRAY" d'un UDT sont représentés comme nouvel élément `<sections>` dans un élément `<member>` dans la structure XML d'un fichier d'importation/exportation. Les membres dans la nouvelle section pour UDT dans un ARRAY sont affectés en tant que sous-éléments avec l'attribut "Path" :

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype="&quot;User_data_type_1&quot;" Remanence="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 &quot;User_data_type_1&quot;">
      <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>
```

Type de données "ARRAY", dans "ARRAY"

Les composants du type de données "ARRAY" dans un autre ARRAY sont représentés comme sous-éléments avec l'attribut "Path" dans la structure XML d'un fichier d'importation/exportation.

Les membres dans un autre ARRAY sont affectés en tant que sous-éléments avec l'attribut "Path" quand le composant est édité par l'utilisateur :

```

<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>

```

Types de données API (UDT)

La structure XML d'un type de données API dépend des paramètres d'exportation réglés dans TIA Portal Openness.

- `ExportOptions.None`
Les membres du type de données API sont écrits seulement quand la valeur par défaut d'un composant au moins est déterminée par l'utilisateur. Pour ces membres, seuls les deux attributs supplémentaires "Name" et "Datatype" sont écrits pour identifier le membre auquel appartient la `<StartValue>`. Les autres membres et attributs ne sont pas écrits.
- `ExportOptions.WithDefaults`
Les attributs suivants sont toujours écrits :
 - Name
 - Datatype
 - ExternalAccessible
 - ExternalVisible
 - ExternalWritable
 - SetPoint
 - StartValue
Ils sont écrits dans XML seulement quand la valeur par défaut dans ce type est déterminée par l'utilisateur. Quand elle est déterminée uniquement dans le type de données API, elle n'est pas écrite.
- `ExportOptions.ReadOnly`
Pour les types de données API, ce paramètre n'entraîne pas un résultat probant. Combiné à d'autres paramètres, il n'a aucune influence sur le résultat.

Variables écrasées

Quand une variable est écrasée par un nouveau type de données, les membres sont représentés dans la structure XML du nouveau type de données. La structure XML ci-dessous montre un type de données WORD qui est écrasé par un ARRAY of BYTE.

```
<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>
```

Interface de bloc

Tous les attributs avec `ReadOnly = "TRUE"` et `Informative = "FALSE"` sont exclus. La structure XML d'une interface de bloc dépend des paramètres d'exportation réglés dans TIA Portal Openness

- `ExportOptions.None`
Avec ce réglage, seules sont exportées les données modifiées ou celles qui n'ont pas leur valeur par défaut.
Quand la définition de l'attribut n'indique pas de valeur par défaut, l'attribut est toujours écrit.
Le fichier d'exportation contient aussi toutes les valeurs obligatoires pour l'importation ultérieure des données.
- `ExportOptions.WithDefaults`
Les attributs suivants sont toujours écrits :
 - `Name`
 - `Datatype`
 - `HmiAccessible` exporté comme `ExternalAccessible`
 - `HmiVisible` exporté comme `ExternalVisible`
 - `ExternalWritable`
 - `SetPoint` (si existant)
 - `Offset` (si existant)
 - `PaddedSize` (si existant)Tous les autres attributs sont écrits seulement quand ils n'ont pas leur valeur par défaut. L'élément `<StartValue` n'est écrit dans XML que s'il a été déterminé explicitement.
- `ExportOptions.ReadOnly`
Pour les interfaces de bloc, ce paramètre n'entraîne pas un résultat probant. Combiné à d'autres paramètres, il n'a aucune influence sur le résultat.

8.4.1.2 Modifications du modèle d'objet et format de fichier XML

Introduction

Pour importer un fichier XML personnalisé ou édité par le biais de TIA PortalOpenness avec succès dans TIA Portal, le fichier doit respecter des schémas définis.

Chaque fichier XML se compose de deux parties principales :

- Interface
- Unité de compilation

La section qui suit propose une description des schémas à respecter par les fichiers.

Interface

Une interface peut contenir plusieurs segments (par ex. Input, InOut, Static) : Vous trouverez tous ces segments dans le répertoire suivant :

C:\Program Files\Siemens\Automation\Portal V14\PublicAPI\V14 SP1\Schemas
\SW.InterfaceSections_v2.xsd

Unité de compilation

Il existe des schémas différents pour les unités de compilation des blocs GRAPH, CONT/LOG et LIST. Vous trouverez les schémas correspondants dans les répertoires suivants :

- GRAPH : C:\Program Files\Siemens\Automation\Portal V ...\PublicAPI\Schemas
\SW.PlcBlocks.Graph.xsd
- CONT/LOG : C:\Program Files\Siemens\Automation\Portal V ...\PublicAPI\Schemas
\SW.PlcBlocks.LADFBD.xsd
- LIST : C:\Program Files\Siemens\Automation\Portal V ...\PublicAPI\Schemas
\SW.PlcBlocks.STL.xsd

Sous-schémas

Il existe les définitions de schéma supplémentaires suivantes, utilisées par toutes les unités de compilation :

- Accès
- Généralités

Accès

Le nœud d'accès décrit par ex. :

- membres locaux/globaux et utilisations constantes
- Appels de FB, FC, d'instructions
- DB pour les appels

Vous trouverez le schéma d'accès dans le répertoire suivant :

C:\Program Files\Siemens\Automation\Portal V14\PublicAPI\V ...\Schemas
\SW.PlcBlocks.Access.xsd

Généralités

Regroupe les attributs et éléments généraux utilisés, par ex. commentaires de toutes sortes, textes et jetons.

Vous trouverez le schéma général dans le répertoire suivant :

C:\Program Files\Siemens\Automation\Portal V14\PublicAPI\V ...\Schemas\SW.Common.xsd

8.4.1.3 Exporter des blocs

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)
- L'API n'est pas en ligne.

Utilisation

L'interface API prend en charge l'exportation de blocs et types de données utilisateur cohérents vers un fichier XML.

Le fichier XML se voit attribuer le nom du bloc. Les types de bloc suivants sont pris en charge :

- Blocs fonctionnels (FB)
- Fonctions (FC)
- Blocs d'organisation (OB)
- Blocs de données globaux (DB)

Les types de langages de programmation suivants sont pris en charge :

- LIST
- LOG
- CONT
- GRAPH
- SCL

Attributs valables pour tous les blocs

Les attributs suivants sont exportés avec les `ExportOptions` sélectionnées dans tous les blocs (voir Exportation de données de configuration (Page 375)). Les attributs représentés en caractères gras sont toujours exportés.

Vous trouverez plus d'informations dans le système d'informations du TIA Portal, sous "Aperçu des attributs du bloc".

Attribut	Type	Valeur par défaut	Protégé en écriture
AutoNumber	Bool	true	false
CodeModifiedDate	DateTime	-	true
CompileDate	DateTime	-	true
CreationDate	DateTime	-	true
HeaderAuthor	String	""	false
HeaderFamily	String	""	false

Attribut	Type	Valeur par défaut	Protégé en écriture
HeaderName	String	""	false
HeaderVersion	String	"0,1"	false
Interface	String	interface non occupée	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	prochain numéro disponible	false
ParameterModified	DateTime	-	true
PLCSimAdvancedSupport	Bool	false	true
ProgrammingLanguage	enum ProgrammingLanguage	-	false
StructureModified	DateTime	-	true

¹ L'attribut IsKnowHowProtected est également valable pour les UDT.

Attributs valables pour les blocs ArrayDB

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs ArrayDB :

Attribut	Type	Valeur par défaut	Protégé en écriture
ArrayDataType	String	-	true
ArrayLimitUpperBound	Int32	-	true

Attributs valables pour les blocs DB

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs DB :

Attribut	Type	Valeur par défaut	Protégé en écriture
IsOnlyStoredInLoadMemory	Bool	false	false
IsPLCDB	Bool	false	false
IsWriteProtectedInAS	Bool	false	false

Attributs valables pour les blocs FB

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs FB :

Attribut	Type	Valeur par défaut	Protégé en écriture
AssignedProDiagFB	String	-	-
ISMultiInstanceCapable	Bool	-	true
Supervisions	String	no supervisions	true pour IDB of FB et false pour FB

Attributs valables pour les blocs DB et FB

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs DB et FB :

Attribut	Type	Valeur par défaut	Protégé en écriture
IsIECCheckEnabled	Bool	false	false
IsRetainMemResEnabled ¹	Bool	false	false
MemoryReserve	Unsigned	0	false
RetainMemoryReserve ²	Unsigned	0	false

² Si la valeur de l'attribut "IsRetainMemResEnabled" est "false" et que l'attribut "RetainMemoryReserve" n'est pas égal à "0", le système déclenche une exception.

Attributs valables pour les blocs FB, blocs de données et IDB

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs FB, blocs de données et IDB :

Attribut	Type	Valeur par défaut	Protégé en écriture
DownloadWithoutReinit	Bool	false	true

Attributs valables pour les blocs FB et FC

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs FB et FC :

Attribut	Type	Valeur par défaut	Protégé en écriture
LibraryType	String	-	true
LibraryTypeVersionGuid	String	-	true

Attributs valables pour les blocs FB et FC (LIST)

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs FB et FC (LIST) :

Attribut	Type	Valeur par défaut	Protégé en écriture
ParameterPassing	Bool	false	false

Attributs valables pour les FB, FC et DB d'instance de blocs FB

Les attributs suivants sont exportés avec les `ExportOptions`. sélectionnées pour FB, FC et DB d'instance de blocs FB :

Attribut	Type	Valeur par défaut	Protégé en écriture
UDABlockProperties	String	""	false
UDAEnableTagReadback	Bool	false	false

Attributs valables pour les DB d'instance de FB et UDT

Les attributs suivants sont exportés avec les `ExportOptions`. sélectionnées pour les DB d'instance de blocs FB et UDT :

Attribut	Type	Valeur par défaut	Protégé en écriture
InstanceOfName	String	""	false
InstanceOfNumber	Unsigned Short	-	true
InstanceOfType	enum BlockType	-	true
OfSystemLibElement	String	""	false
OfSystemLibVersion	String	""	false

Attributs valables pour les blocs OB

Les attributs suivants sont exportés avec les `ExportOptions`. sélectionnées pour les blocs OB pour des API Plus spécifiques :

Attribut	Type	Valeur par défaut	Protégé en écriture
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
EnableTimeError	Bool	-	true

8.4 Importation/exportation de données d'un appareil API

Attribut	Type	Valeur par défaut	Protégé en écriture
EventClass	String	-	true
EventsToBeQueued	Int32	-	true
EventThresholdForTimeError	Int32	-	true
Execution	OExecution	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 Lors de l'exportation d'un OB, le SecondaryType est défini par ailleurs à l'aide du numéro d'OB. L'affectation est vérifiée pendant le processus d'importation. Si l'affectation est incorrecte, une exception de type "Recoverable" est déclenchée.

Attributs valables pour les blocs FB, FC et OB

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs FB, FC et OB :

Attribut	Type	Valeur par défaut	Protégé en écriture
HandleErrorsWithinBlock	Bool	false	true

Attributs valables pour les blocs FB, FC et UDT

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs FB, FC et UDT :

Attribut	Type	Valeur par défaut	Protégé en écriture
LibraryConformanceStatus	String	-	false

Attributs valables pour les blocs GRAPH

Les attributs suivants sont exportés avec les `ExportOptions` . sélectionnées pour les blocs GRAPH :

Attribut	Type	Valeur par défaut	Protégé en écriture
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

Attributs valables pour les blocs GRAPH FB

Les attributs suivants sont exportés avec les `ExportOptions`. sélectionnées pour les blocs GRAPH FB :

Attribut	Type	Valeur par défaut	Protégé en écriture
WithAlarmHandling	Bool	true	false

Attributs valables pour les blocs SCL

Les attributs suivants sont exportés avec les `ExportOptions`. sélectionnées pour les blocs SCL. Ces attributs sont exportés sur la base du type des API.

Attribut	Type	Valeur par défaut	Protégé en écriture
CheckArrayLimits	Bool	false	false
ExtendedStatus	Bool	false	false
DBAccessibleFromOPCUA	Bool	true	false

Attributs valables pour les blocs GRAPH, SCL et CONT/LOG

Les attributs suivants sont exportés avec les `ExportOptions`. sélectionnées pour les blocs GRAPH, SCL et CONT/LOG :

Attribut	Type	Valeur par défaut	Protégé en écriture
SetENOAutomatically	Bool	-	false

Code de programme

Pour exporter un bloc dépourvu de protection Know-how vers un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

8.4.1.4 Exporter des blocs avec protection know-how

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)
- L'API n'est pas en ligne.

Utilisation

Le fichier XML qui en résulte ressemble au fichier d'exportation d'un bloc sans protection know-how. L'exportation couvre toutefois les données visibles de l'interface utilisateur lorsque le bloc s'ouvre sans mot de passe.

La liste d'attributs du bloc indique que le bloc correspondant possède une protection know-how.

Code du programme

Pour exporter les données visibles d'un bloc doté d'une protection know-how vers un fichier XML, modifiez le code de programme suivant :

```
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);
}
```

8.4.1.5 Exportation/importation de blocs SCL

Instructions SCL avec balises XML pour l'export

L'opération d'exportation pour les blocs SCL effectue l'exportation de vos balises XML correspondantes sur la base du type d'instructions SCL. Cette opération prend en charge les réseaux SCL d'instructions SCL dans des blocs LAD/FBD d'instructions SCL. Les instructions SCL sont classées en tant qu'éléments de textes, d'opérandes, d'expressions, d'instructions de commande, etc. Les instructions de bloc SCL avec leurs balises XML et attributs exportés correspondants sont indiquées ci-dessous.

Nouvelle ligne

De nouvelles lignes dans les blocs SCL sont représentées par des balises NewLineXML.

- Contient l'attribut non signé Num avec la valeur par défaut 1.
- L'attribut Num n'est pas réglé sur la valeur 0.
- Pris en charge uniquement pour SCL.

Bloc SCL	Balise XML
	<NewLine Num="2" />

Vide

Les espaces dans les blocs SCL sont représentés par des balises BlankXML.

- Contient l'attribut non signé Num avec la valeur par défaut 1.
- L'attribut Num n'est pas réglé sur la valeur 0.
- Pris en charge uniquement pour SCL.
- Ne prend pas en charge l'attribut Integer, qui est disponible dans les autres langages de STEP 7.

Bloc SCL	Balise XML
	<Blank Num="2" />

Indentation d'instructions de bloc SCL

Dans les paramètres du TIA Portal, vous pouvez modifier l'indentation du code SCL via Options/Settings/General/Script/text editors. Le type d'indentation est défini dans le tableau suivant, sur la base du mode d'identification.

Mode d'identification	Résultat
Sans	L'opération d'importation insère les espaces comme indiqués dans les fichiers source.
Paragraphe ou Smart	L'opération d'importation insère les espaces d'identification indiqués dans le fichier importé.

Sur la base de l'indentation sélectionnée, celle-ci est effectuée dans le XML file importé du bloc SCL.

Commentaire

Les commentaires sur ligne unique et plusieurs lignes dans des blocs SCL sont représentés par des balises XML LineComment.

- Seule la balise LineComment (pour commentaire unilingue) est prise en charge par SCL.
- La balise Comment (pour commentaire multilingue) n'est pas prise en charge par SCL.
- Contient l'attribut Inserted avec la valeur par défaut false.
- Inserted="false" affiche un commentaire "/" sur une seule ligne dans les blocs SCL .
- Inserted="true" affiche un commentaire "(**)" sur plusieurs lignes dans les blocs SCL.
- NoClosingBracket="true" affiche des commentaires sans parenthèses de fermeture dans des blocs SCL. Cet attribut est facultatif et a la valeur par défaut false.
- XML n'indique aucune hiérarchie de commentaires dans les blocs SCL.

Bloc SCL	Balise XML
// one line comment	<LineComment> <Text>one line comment</Text> </LineComment>
(* one line comment second line *)	<LineComment Inserted="true"> <Text>one linecomment secondline</Text> </LineComment>
(* first comment (* second comment *) end first comment *)	<LineComment Inserted="true"> <Text> first comment (* second comment *) end first comment</ Text> </LineComment > Le commentaire imbriqué fait partie du texte de commentaire externe.
(* comment without closing bracket	<LineComment Inserted="true" NoClosingBracket="true"> <Text> comment without closing bracket</Text> </LineComment >

Région

Les régions dans les blocs SCL sont représentées par des balises Token XML.

- La balise Text XML représente le nom de la région (region_name).
- La casse (majuscule et minuscule) n'a pas d'importance pour l'attribut Text de la balise Token XML.

- La casse est prise en compte lors de la procédure d'importation et l'éditeur affiche les mots-clés tels qu'ils sont configurés dans les paramètres de TIA Portal.
- Si le mot-clé end_region se termine par ";" (point-virgule) dans le bloc SCL, le caractère ";" est présent dans la balise Text XML.

Bloc SCL	Balise XML
<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 dans les blocs SCL sont représentés sous forme de balise Token XML. Les paramètres sont affichés dans la balise XML Access avec l'attribut Scope en tant que LiteralConstant.

Bloc SCL	Balise XML
<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>

Constantes : Constantes littérales

Les constantes dans les blocs SCL sont représentées par des balises AccessXML.

- L'attribut Scope peut être réglé sur des valeurs telles que LiteralConstant, TypedConstant, LocalConstant, et GlobalConstant..
- Les noms des constantes qui portent le caractère "#" en préfixe sont ignorés dans XML.
- Le caractère "#" est ajouté lors du processus d'importation de XML.
- La valeur des constantes globales, représentée par des guillemets, est ignorée dans XML.
- Les guillemets sont ajoutés lors du processus d'importation de XML.

8.4 Importation/exportation de données d'un appareil API

Type de constante	Bloc SCL	Balise XML
Constante littérale : Nombre entier	#Out := 10;	<pre><Access Scope="LiteralConstant"> <Constant> <ConstantValue>10</ConstantValue> <ConstantTypeInformative="true">LINT</ ConstantType> </Constant> </Access></pre>
Constante littérale : String	#myString := 'Hello world';	<pre><Access Scope="LiteralConstant"> <Constant> <ConstantValue>Hello world</ ConstantValue> <ConstantTypeInformative="true">STRING</ ConstantType> </Constant> </Access></pre>
Constante littérale : Typé	#Out := int#10;	<pre><Access Scope="TypedConstant"> <Constant> <ConstantValue>int#10</ConstantValue> </Constant> </Access></pre>
		<p>Format de XML après exportation avec réglage ExportOptions.ReadOnly.</p> <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>
Constante locale	#Out := #mylocal;	<pre><Access Scope="LocalConstant"> <Constant Name="mylocal" /> </Access></pre>
		<p>Format de XML après exportation avec réglage ExportOptions.ReadOnly</p> <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>

Type de constante	Bloc SCL	Balise XML
Constante globale	#Out := "myglobal";	<Access Scope="GlobalConstant"> <Constant Name="myglobal" /> </Access>
		<p>Format de XML après exportation avec réglage ExportOptions.ReadOnly.</p> <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>

Les constantes d'adresse ne sont pas prises en charge dans les blocs SCL et sont ignorées dans ce tableau.

Variables

Les variables locales et globales dans les blocs SCL sont représentées par des balises Access XML.

- L'attribut Scope possède les valeurs de LocalVariable et de GlobalVariable.
- La balise XML permettant d'attribuer la valeur 10 est ignorée ici.

Type de variable	Bloc SCL	Balise XML
Variable locale	#Out := 10;	<Access Scope="LocalVariable"> <Symbol> <Component Name="Out" /> </Symbol> </Access>
Variable globale	"Tag_3" := 10;	<Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_3" /> </Symbol> </Access>
		<p>Format de XML après exportation avec réglage ExportOptions.ReadOnly.</p> <pre><Access Scope="GlobalVariable"> <Symbol> <Component Name="Tag_3" /> <Address Area="Memory" Type="Int" BitOffset="96" Informative="true" /> </Symbol> </Access></pre>

Expressions

Les expressions simples dans les blocs SCL sont représentées par les balises Access XML. L'attribut Scope possède la valeur de LocalVariable pour les expressions.

Bloc SCL	Balise XML
#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>

Structures de commande dans les blocs SCL

Les instructions de commande telles que IF, CASE, FOR, WHILE, REPEAT, GOTO, EXIT, CONTINUE, and RETURN sont représentées par des balises Token XML .

- Les symboles conditionnels utilisés dans le bloc SCL, tels que >, <, &, sont représentés dans XML sous forme de séquences d'échappement (< > &).
- Cette combinaison de balises XML n'est valide que pour des blocs SCL. Une exception sera déclenchée dans les autres langages.

Nom du bloc	Bloc SCL	Balise XML
IF	IF #a<#c THEN ; END_IF;	<Token Text="IF" /> <Blank /> <Access Scope="LocalVariable"> <Symbol> <Component Name="a" /> </Symbol> </Access> <Token Text="<" /> <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=";" />

8.4 Importation/exportation de données d'un appareil API

Nom du bloc	Bloc SCL	Balise XML
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> <Tok en 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 > <NewLine /></pre>

8.4 Importation/exportation de données d'un appareil API

Nom du bloc	Bloc SCL	Balise XML
		<pre> <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>
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>

Nom du bloc	Bloc SCL	Balise XML
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>

8.4 Importation/exportation de données d'un appareil API

Nom du bloc	Bloc SCL	Balise XML
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>

Nom du bloc	Bloc SCL	Balise XML
GOTO	<pre> here // well : // this is goto statement </pre>	<p>Exemple XML pour la définition d'étiquettes GOTO</p> <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>	<p>Exemple XML pour l'utilisation d'étiquettes GOTO</p> <pre> <Token Text="GOTO" /> <Blank /> <LineComment inserted="true"> <Text>comment</Text> </LineComment> <Blank /> <Access Scope="Label"> <Label Name="here" /> </Access> <Token Text=";" /> </pre>

Attributs de référencement

Les attributs de référencement de blocs SCL sont représentées par l'attribut AccessModifier de la balise Component.

- Pour la référence simple, AccessModifier est réglé sur la valeur Reference.
- Pour la référence Array, AccessModifier est réglé sur la valeur ReferenceToArray..

Bloc SCL	Balise XML
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>

8.4.1.6 Exportation/importation de types structurés de blocs SCL

Types SCL structurés avec des balises XML pour l'exportation

Dans les types SCL structurés, vous pouvez insérer des espaces, de nouvelles lignes et des commentaires aux instructions SCL. Les instructions SCL structurées avec leurs balises XML et attributs exportés correspondants sont indiqués ci-dessous.

Accès global

Dans les instructions SCL, les variables et des constantes pour l'accès global sont mises entre guillemets. Les commentaires entre les variables et des parties d'adresse sont représentés par la balise LineComment XML.

Bloc SCL	Balise XML
<pre>"Bloc de données_1".(*Commentaire 1*)Statique_1(*Commentaire 2*).Statique_2</pre>	<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>
<pre>"Data_block_1".Static_1 := 10</pre>	<p>Format de XML après exportation avec réglage ExportOptions.None</p> <pre><Access Scope="GlobalVariable"> <Symbol> <Component Name="Data_block_1" /> <Token Text="." /> <Component Name="Static_1" / > </Symbol> </Access></pre> <p>Format de XML après exportation avec réglage ExportOptions.ReadOnly</p> <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>

Utilisation de guillemets et de

Les guillemets utilisés au premier niveau décrivent le type de variable et font office de séquences Escape pour les caractères spéciaux dans les instructions SCL. Si les guillemets sont utilisés dans le premier niveau, définissez la variable en tant que variable globale. Si les guillemets sont utilisés après #, ils représentent la séquence d'échappement de caractères spéciaux comme # ou d'espaces.

- Dans le fichier XML, la balise BooleanAttributes avec l'attribut Name est utilisée pour représenter l'utilisation différente. Le Name contient des valeurs telles que HasQuotes et HasHash.
- # est spécifié pour définir la structure dans l'attribut d'étendue.
- Ces valeurs s'appliquent uniquement à SCL.
- Les valeurs par défaut pour ces balises sont initialement FALSE, mais les valeurs ne sont jamais exportées, y compris avec le paramètre ExportOptions.WithDefaults.

Bloc SCL	Balise XML
"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>

Tableau

SCL permet l'insertion de commentaires dans les indices de tableau avec "[" et "]". Dans le fichier XML, l'attribut AccessModifier dans la balise Component sert à marquer la présence de tableaux.

- Si Accessmodifier comporte la valeur Array, une balise Access subordonnée est obligatoire afin d'indiquer la variable d'indice du tableau.
- La valeur par défaut de AccessModifier est None.

Bloc SCL	Balise XML
#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>

Accès absolu

SCL permet différents types d'accès, par ex. l'accès absolu, décalage absolu, mixte (base de données et variable membre), par tranche, périphérique et direct. L'accès absolu est représenté dans XML par la balise Address.

- Le signe % dans DB n'est pas écrit dans XML. Il est créé automatiquement lors de l'importation.
- Des espaces sont autorisés entre les parties de l'adresse.

Bloc SCL	Balise XML
%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;	<p>L'instruction XML suivante s'applique à tous les langages sauf SCL.</p> <pre><Access Scope="Address"> <Address Area="DB" BlockNumber="20" BitOffset="83" Type="Bool" /> </Access></pre> <p>L'instruction XML suivante s'applique à SCL.</p> <pre><Access Scope="Address"> <Symbol> <Address Area="DB" BlockNumber="20" /> <Token Text="." /> <Address Area="DB" BitOffset="83" Type="Bool"/> </Symbol> </Access></pre>

Type d'accès "décalage absolu"

Dans LIST, la balise AbsoluteOffset représente le type d'accès "Décalage absolu". Dans SCL, la balise Address est utilisée pour l'accès absolu.

Bloc SCL	Balise XML
#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>

Type d'accès "en tranche" (slice)

Dans SCL, l'attribut SliceAccessModifier n'est pas pris en charge, et le type d'accès "en tranche" est représenté par la balise Token.

Bloc SCL	Balise XML
"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>

Accès périphérique

L'accès périphérique est représenté par la balise Token.

Bloc SCL	Balise XML
"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>

Type d'accès "direct"

Les instructions TypeOf et TypeOfDB sont exécutées soit avec un type défini par le système, soit avec un type défini par l'utilisateur. Les types sont indiqués dans la balise Access par l'attribut Scope et les valeurs SystemType et UserType.

Bloc SCL	Balise XML
<p>Exemple de type défini par le système</p> <pre>if TypeOf(#inVariant) = TO_SpeedAxis then ... end_if</pre>	<pre><Token text="=" /> </Blank> <Access Scope="SystemType"> <DataType>TO_SpeedAxis</DataType> </Access></pre>
<p>Exemple de type défini par l'utilisateur</p> <pre>if TypeOf(#inVariant) = "aUserDefinedType" then ... end_if</pre>	<pre><Token text="=" /> </Blank> <Access Scope="UserType"> <DataType>aUserDefinedType</ DataType> </Access></pre>

8.4.1.7 Exportation/importation de blocs d'appel SCL

Blocs d'appel SCL avec balises XML pour l'exportation

Les paramètres d'appel SCL sont représentés dans XML par la balise Parameter. L'attribut informative est utilisé pour représenter les paramètres FB non-affectés et les valeurs en retour telles que l'horodatage, les informations de memento, etc. Le format XML suit le même ordre arbitraire que dans le bloc SCL.

Voici un exemple d'appel de bloc :

Bloc SCL	Balise XML
<pre>#Callee_Instance(Input_1 := 5);</pre>	<p>Format de XML après exportation avec réglage ExportOptions.None</p> <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>
	<p>Format de XML après exportation avec réglage ExportOptions.ReadOnly</p> <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>

Bloc SCL	Balise XML
	<pre><Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Int</ ConstantType> <ConstantValue>5</ ConstantValue> </Constant> </Access> </Parameter> <Token text=")" /> </CallInfo> </Access> <Token text=";" /></pre>

Exemple de paramètres indépendants

Le FB possède quatre paramètres, parmi lesquels a, b, c ainsi que d. b et d ne sont pas connectés.

Bloc SCL	Balise XML
<pre>"Block_4_DB" (a:=TRUE,c:=TRUE) ;</pre>	<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>

Exemple pour "un paramètre"

Le bloc SCL permet l'omission du nom du paramètre. Ce paramètre est représenté par la balise NamelessParameter. La balise NamelessParameter ne possède pas d'attributs et s'applique uniquement à SCL.

Bloc SCL	Balise XML
<pre>"Block_4_DB" (TRUE);</pre>	<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>

Expression sous forme de paramètre effectif

Bloc SCL	Balise XML
#Callee_Instance(Input_1 := #a+3);	<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=";" />

Expression sous forme de paramètre effectif sans paramètre formel

Bloc SCL	Balise XML
#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>

Appel de fonction

Bloc SCL	Balise XML
#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>

Appel absolu

Dans SCL, l'appel peut être lancé via l'adresse absolue du DB. L'attribut Name du nœud CallInfo est vide en raison de l'adresse absolue.

Une exception récupérable est déclenchée par l'importation si

- un nœud "Address" avec une valeur valide de l'attribut Name est disponible ;
- le nœud "Address" n'est pas disponible et n'est pas une valeur valide de l'attribut Name.

Bloc SCL	Balise XML
<code>%DB20 (...);</code>	<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>

Instruction

L'instruction dans le bloc SCL est vérifiée dans la bibliothèque système lors du processus d'importation et les versions d'instructions ne sont pas exportées lors du processus d'exportation.

Le type d'instruction général est donné ci-dessous.

Bloc SCL	Balise XML
<pre>#myInt := ATTACH(OB_NR := 1, EVENT := 15, ADD := TRUE);</pre>	<p>Format de XML après exportation avec réglage ExportOptions.ReadOnly</p> <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>

Bloc SCL	Balise XML
	<pre></Constant> </Access> </Parameter> <Parameter Name="RET_VAL" Informative="true" /> <Token text=")" /> </Instruction> </Access> <Token text=";" /></pre>

Instruction avec modèle

Si le paramètre de modèle complète les noms d'instruction, l'exportation du paramètre de modèle est nécessaire. Lorsqu'une balise "TemplateValue" avec l'attribut Type="Type" suit la balise Instruction, le processus d'importation enchaîne la valeur de modèle avec le nom de l'instruction.

Bloc SCL	Balise XML
<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>

Conversion

Avec les fonctions de conversion, le vrai nom de l'instruction et son modèle ne sont pas exportés. Au lieu de cela, le nom utilisé dans le bloc SCL est exporté.

Bloc SCL	Balise XML
<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>

Instruction avec instance

L'instance et l'instruction sont séparées par des espaces. Les espaces sont facultatifs, et ils peuvent être représentés par de nouvelles lignes et de nouveaux commentaires. L'instruction TON est représentée par l'attribut Name de la balise Instruction.

Bloc SCL	Balise XML
<pre>IEC_Timer_0_DB . TON (IN:="Tag_1", PT:="Tag_2");</pre>	<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>

Constante d'alarme

Les constantes d'alarme ne sont utilisées que dans S7 400 PLCs, et le code XML exporté est similaire à d'autres langages.

Bloc SCL	Balise XML
"Block_1_DB"(16#0000_0001);	<p>Format de XML après exportation avec réglage ExportOptions.None</p> <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> <p>Format de XML après exportation avec réglage ExportOptions.ReadOnly</p> <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 (sortie de validation)

Afin de prendre en charge la sortie de validation ENO dans le bloc SCL, l'attribut "Scope" est utilisé avec la valeur "PredefinedVariable" dans la balise "Access". Il contient également la balise "PredefinedVariable" en tant qu'élément subordonné de la balise Access.

- La balise "PredefinedVariable" possède un attribut obligatoire "Name".
- Le volume "PredefinedVariable" et la balise "PredefinedVariable" ne sont autorisés que pour SCL.

Bloc SCL	Balise XML
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>

8.4.1.8 Exporter des blocs F

Exporter des blocs de sécurité

Les blocs de sécurité sont exportés comme blocs standard. Pour les blocs de sécurité, la valeur de l'attribut "ProgrammingLanguage" commence par un préfixe "F_".

Remarque

L'importation d'un fichier est impossible si la valeur de l'attribut "ProgrammingLanguage" commence par un préfixe "F_".

Importer des blocs de sécurité comme blocs standard

Les blocs de sécurité peuvent être importés comme des blocs standard si le préfixe "F_" est supprimé dans la valeur de tous les attributs "ProgrammingLanguage".

Voir aussi

Etablissement d'une connexion au portail TIA (Page 74)

Ouvrir un projet (Page 99)

Exporter des blocs (Page 448)

Exporter des blocs avec protection know-how (Page 454)

8.4.1.9 Exporter des blocs système

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)
- Le projet contient un bloc système.
- L'API n'est pas en ligne.

Utilisation

Seuls les blocs système visibles sont disponibles dans la composition des blocs, donc pas de SFB ni de SFC. Le fichier XML qui en résulte ressemble au fichier d'exportation d'un bloc.

Code du programme

Pour exporter les données visibles d'un bloc vers un fichier XML, modifiez le code de programme suivant :

```
//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);
        }
    }
}
```

8.4.1.10 Exporter des blocs GRAPH avec texte multilingue

Structure XML de blocs GRAPH avec texte multilingue

Le fichier d'exportation XML de blocs GRAPH contient les noms d'étapes et de transitions compilés du graphe. Ces textes multilingues traduits sont représentés respectivement sous les éléments de niveau supérieur Step et Transition comme éléments StepName et TransitionName. Ces éléments contiennent un élément MultiLanguageText pour chaque langue prise en charge. Les textes dans des langues non paramétrées de manière explicite ne sont pas exportés. Si aucune traduction n'est disponible, les éléments StepName et TransitionName ne sont pas exportés. Les éléments StepName et TransitionName sont optionnels. L'opération d'importation XML depuis Openness déclenche une exception récupérable pour les versions de Graph antérieures à V5.0.

Exemple pour l'élément 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>
```

Exemple pour l'élément 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>
```

8.4.1.11 Importer un bloc

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Utilisation

La TIA Portal Openness API prend en charge l'importation de blocs avec les langages de programmation "LIST", "LOG", "GRAPH", "SCL" ou "CONT" à partir d'un fichier XML. Les types de bloc suivants sont pris en charge :

- Blocs fonctionnels (FB)
- Fonctions (FC)
- Blocs d'organisation (OB)
- Blocs de données globaux (DB)

Remarque

Importation de blocs de données optimisés

Les blocs de données optimisés sont uniquement pris en charge par les CPU à partir de S7-1200. Si vous importez des blocs de données optimisés dans une S7-300 ou S7-400, une exception est déclenchée et l'importation échoue.

Réaction à l'importation

Pour l'importation d'un bloc, les règles à respecter sont les suivantes :

- Le fichier XML peut contenir moins de données que le bloc se trouvant dans le projet, par ex. moins de paramètres.
- Les informations redondantes telles que les informations d'appel doivent être identiques dans le projet et dans le fichier XML. Faute de quoi, une exception est lancée.
- Les données du fichier XML peuvent être "incohérentes" en ce qui concerne leur capacité à être compilées dans TIA Portal.
- Les attributs possédant les attributs "ReadOnly=True" et "Informative=True" ne sont pas importés.
- Les DB d'instance manquants ne sont pas automatiquement créés.
- Si aucun numéro de bloc n'est indiqué dans le fichier XML, il est automatiquement attribué.
- Si le bloc ne figure pas dans le projet et si aucune information de version n'est indiquée dans le fichier XML, la version "0.1" est attribuée.

Code de programme

Modifiez le code de programme suivant :

```
//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);
}
```

Modifiez le code de programme suivant :

```
//Import system blocks
private static void ImportSystemBlocks(PlcSoftware plcSoftware)
{
    PlcBlockSystemGroup systemblockGroup = plcSoftware.BlockGroup;
    IList<PlcBlock> blocks = systemblockGroup.Blocks.Import(new FileInfo(@"D:\Blocks
\myBlock.xml"), ImportOptions.Override);
}
```

8.4.2 Tables des variables

8.4.2.1 Exporter des tables de variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Utilisation

Un fichier XML est exporté par table des variables API.

L'interface TIA Portal Openness API prend en charge l'exportation de toutes les tables de variables API du groupe système et de ses sous-groupes.

Code du programme

Pour exporter toutes les tables de variables API du groupe système et de ses sous-groupes, modifiez le code de programme suivant :

```
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);
    }
}
```

Voir aussi

Exportation de données de configuration (Page 375)

8.4.2.2 Importer une table de variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Code du programme

Pour importer des tables de variables API ou une structure de dossiers avec des tables de variables API dans le groupe système ou un groupe personnalisé depuis un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

Voir aussi

Remarques sur la performance de TIA Portal Openness (Page 41)

8.4.2.3 Exporter des variables ou constantes individuelles d'une table de variables API

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Utilisation

L'interface API prend en charge l'exportation d'une variable ou d'une constante depuis une table de variables API vers un fichier XML. Ce faisant, veillez à ce que les noms utilisés pour les tables des variables correspondent aux conventions de dénomination de fichiers de votre système de fichiers.

Le commentaire d'une variable ou d'une constante n'est exporté que si au moins une langue est définie pour le commentaire. Si le commentaire n'est pas défini pour toutes les langues du projet, il est uniquement exporté pour les langues du projet définies.

Remarque

Constantes système API

Les constantes système API sont exclues de l'exportation et de l'importation.

Code de programme

Pour exporter une variable ou une constante déterminée d'une table de variables API vers un fichier XML, modifiez le code de programme suivant :

```
//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) return;

    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) return;

    plcConstant.Export(new FileInfo(string.Format(@"D:\Samples\{0}.xml",
plcConstant.Name)), ExportOptions.WithDefaults);
}
```

Voir aussi

Exportation de données de configuration (Page 375)

Remarques sur la performance de TIA Portal Openness (Page 41)

8.4.2.4 Importer une seule variable ou constante dans une table de variables API

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal (Page 74)
- Un projet est ouvert.
Voir Ouverture d'un projet (Page 99)

Utilisation

Lors d'un appel d'importation, vous pouvez importer soit des variables soit des constantes.

Remarque

Les constantes peuvent uniquement être importées comme constantes utilisateur.

Code du programme

Pour importer des groupes de variables ou certaines variables ou constantes depuis un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

Voir aussi

[Exportation de données de configuration \(Page 375\)](#)

[Remarques sur la performance de TIA Portal Openness \(Page 41\)](#)

8.4.3 Exporter un type de données utilisateur**Conditions requises**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Code de programme

Pour exporter un type de données utilisateur vers un fichier XML, modifiez le code de programme suivant :

```
//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);
}
```

8.4.4 Importer un type de données utilisateur

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Utilisation

L'interface API prend en charge l'importation de types de données utilisateur à partir d'un fichier XML.

Syntaxe du fichier d'importation

L'exemple de code suivant présente un extrait d'un fichier d'importation d'un type de données utilisateur :

```
<Section Name="Input">
  <Member Name="Input1" Datatype="myudt1">
    <Sections>
      <Section Name="None">
        <Member Name="MyUDT1Member1" Datatype="bool"/>
        <Member Name="MyUDT1Member2" Datatype="myudt1">
          <Sections...>
```

Remarque**Syntaxe pour les types de données personnalisés d'éléments**

Si le type de données personnalisé d'un élément présente une mauvaise syntaxe dans le fichier d'importation pour types de données utilisateur, une exception est déclenchée.

Assurez-vous que `"` est utilisé pour noter les types de données personnalisés.

Code du programme

Pour importer un type de données utilisateur, modifiez le code de programme suivant :

```
//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);
}
```

Voir aussi

Importation de données de configuration (Page 377)

8.4.5 Exportation de données au format OPC UA XML**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API n'est pas en ligne.

Utilisation

Vous pouvez exporter des données API dans un fichier au format OPC UA XML en appelant une action sur la TIA Portal Openness API. En paramètre d'entrée de l'action, vous avez besoin d'un chemin de répertoire absolu dans lequel sera enregistré le fichier XML.

Code de programme

Modifiez le code de programme suivant :

```
//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)));
}
```

8.5 Importation/exportation de données matérielles

8.5.1 Format de fichier AML

Introduction

AutomationML est un format de données neutre basé sur XML pour l'enregistrement et l'échange d'informations d'ingénierie d'une installation et qui est proposé sous forme de standard ouvert. L'objectif d'AutomationML est la connexion du paysage hétérogène des outils d'ingénierie modernes dans leurs différentes disciplines telles que la conception mécanique d'installations, la conception électrique, IHM, API, la commande de robots.

Le modèle de classification utilisé pour l'exportation et l'importation de données CAx est basé sur les normes AML suivantes :

- Whitepaper AutomationML partie 1 – AutomationML Architecture, octobre 2014
- Whitepaper AutomationML partie 2 – AutomationML Architecture, octobre 2014
- Whitepaper AutomationML partie 4 – AutomationML Logic, mai 2010
- Whitepaper AutomationML – AutomationML Communication, septembre 2014
- Whitepaper AutomationML – AutomationML and eCl@ss Integration, novembre 2015
- Recommandations d'utilisation : Automation Project Configuration - AR_001E Version 1.0.0, Mai 2017

Schéma

Le modèle d'échange de données AutomationML est décrit par le schéma CAEX, version 2.15. Vous pouvez télécharger ce fichier sur la page d'accueil AutomationML e.V. (<https://www.automationml.org/o.red.c/dateien.html>)

8.5.2 Pruned AML

Introduction

Pruning (élagage) décrit le processus d'optimisation du contenu par la suppression de certains éléments qui ne doivent pas nécessairement être indiqués. Avec des outils externes comme EPLAN, les informations de sous-modules créées automatiquement n'ont aucune signification quant à EPLAN dans une configuration matérielle. C'est pourquoi ces outils génèrent un fichier AML en supprimant de la configuration matérielle les informations de sous-modules créées automatiquement. Ce fichier est appelé Pruned AML.

Génération du fichier Pruned AML

La génération de Pruned AML est basée sur les règles suivantes dans cet ordre.

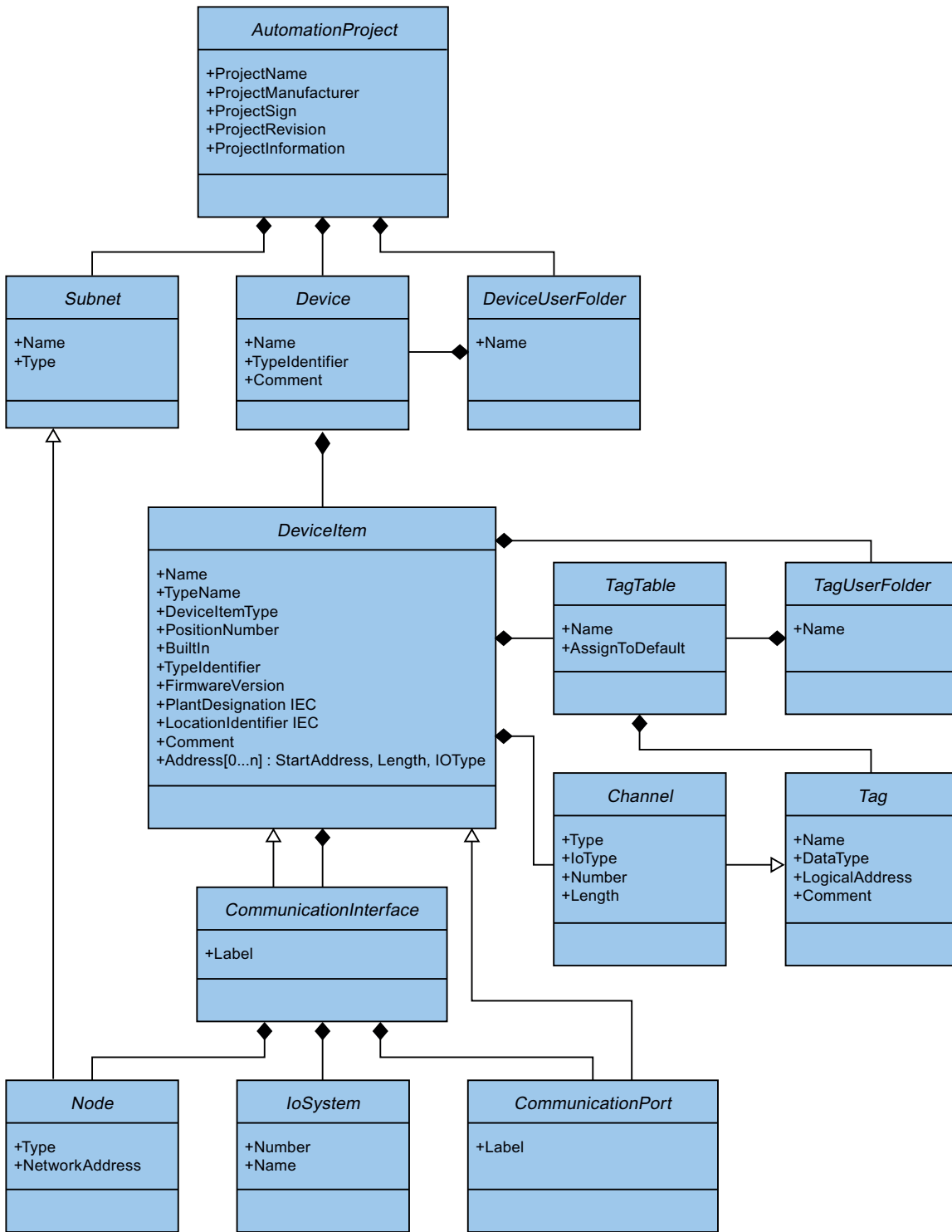
1. Si un élément d'appareil est enfichable, aucun Pruning n'est effectué.
2. Si un élément d'appareil est de type "Interface" ou "Port", aucun Pruning n'est effectué.
3. Les objets d'adresse de type "Diagnostic" ne sont pas pertinents pour l'algorithme de Pruning.
4. Les objets d'adresse reliés aux sous-modules créés automatiquement sont indiqués sous l'élément directement supérieur (qui ne doit pas être un sous-module créé automatiquement).
5. Les objets d'adresse sont inclus dans la même séquence que celle affichée par TIA Portal Openness.

8.5.3 Vue d'ensemble des objets et paramètres de l'importation/exportation CAx

Objets et attributs de l'exportation/importation

La figure suivante représente les objets exportables avec leurs attributs et dépendances dans l'importation/exportation CAx.

8.5 Importation/exportation de données matérielles



8.5.4 Structure des données CAx pour l'importation/exportation

Structure de base d'un fichier d'exportation

Les données du fichier d'exportation issues de l'importation/exportation sont organisées au moyen d'une structure de base. Le fichier d'exportation est créé au format AML.

Le fichier AML commence par une information sur le document. Ce fichier comporte les données de l'installation spécifique à l'ordinateur du projet exporté.

Le fichier d'exportation comprend les deux zones suivantes :

- Pour plus d'informations...

<WriterHeader> contient des informations sur le processus d'exportation ou d'importation. L'importation ignore le contenu de la zone <AdditionalInformation>. Vous pouvez par ex. insérer un bloc avec <AdditionalInformation>...</AdditionalInformation> et y placer des informations supplémentaires à la validation. Après la transmission du fichier AML, le destinataire peut vérifier avant l'importation avec ce bloc si le fichier AML a été modifié.

```
<xml version="1.0" encoding="utf-8">
<CAEXFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  FileName="CAx_asterisk_AML_03_V14.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>1400</WriterVersion>
      <WriterRelease>1400.100.101.16</WriterRelease>
      <LastWritingDateTime>2016-09-29T11:21:34.9551066Z</LastWritingDateTime>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0" />
  <AdditionalInformation DocumentVersions="Recommendations">
    <Document DocumentIdentifier="AR APC" Version="1.0.0" />
  </AdditionalInformation>
  ...
```

- Hiérarchie d'instance

Cette zone contient l'ordre hiérarchique des éléments internes exportés.

```

...
<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>

```

Éléments internes

Tous les objets dans la hiérarchie d'instance du fichier AML sont `InternalElements`. L'élément interne `AutomationProject` contient tous les éléments internes de toutes les classifications de rôles. Chaque élément interne prend en charge un ensemble d'attributs.

L'attribut `<TypeIdentifier>` identifie chaque type d'objet d'un objet matériel pouvant être créé via TIA Portal Openness.

Remarque

Objets automatiquement créés

Les objets automatiquement créés peuvent uniquement être créés par d'autres objets. Ils n'ont aucun attribut ni aucun identifiant de type. Ils sont inclus dans le fichier exporté mais vous ne pouvez pas lancer l'exportation d'un objet spécifique automatiquement créé.

À la fin de l'élément AML d'un élément interne, ce qui suit est défini :

- Classification des rôles
L'élément `SupportedRoleClass` définit le type d'objet d'un élément interne. Le type d'objet est défini dans la bibliothèque de classification des rôles qui représente la norme AML pour le modèle d'objet TIA Portal Openness et TIA Portal.

```
...
<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>
</InternalElement>
...
```

- Liens internes
L'élément `InternalLink` définit les partenaires de communication d'une connexion.

```
...
<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>
```

Attributs

Les attributs sont affectés à des éléments internes comme suit :

```

...
<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>

```

Manipulation des attributs

La manipulation d'attributs est définie comme suit pour chaque attribut :

- Ignoré
L'attribut est ignoré lors de l'importation et n'est pas présent dans le fichier d'exportation.
- Obligatoire
L'attribut doit être présent dans un fichier d'importation et ne peut pas être supprimé du fichier d'exportation.
- Optionnel
Si cet attribut est absent du fichier d'exportation, sa valeur par défaut est définie. Cet attribut est absent dans le fichier d'exportation s'il n'est pas utilisable pour un objet, par ex. si les modules n'ont pas tous une version de firmware.

8.5 Importation/exportation de données matérielles

- Exporter seulement
L'attribut, tel que le nom de type de l'élément de l'appareil, est défini en interne par TIA Portal. Si celui-ci est présent dans un fichier d'importation, il est ignoré par TIA Portal lors de l'importation.
- Importer seulement
L'attribut peut influencer sur le comportement de l'importation. Si cet attribut est absent du fichier d'importation, le comportement correspondra à la valeur par défaut de l'attribut.

Voir aussi

Identifiants de type AML (Page 512)

8.5.5 Identifiants de type AML

Éléments internes

La chaîne `TypeIdentifieur` est constituée des parties suivantes :

- `<TypeIdentifieurType>:<Identifieur>`

Les valeurs suivantes de `TypeIdentifieurType` sont admissibles ici :

- `OrderNumber` sert à indiquer des modules physiques
- `GSD` sert à indiquer des appareils basés sur GSD/GSDML
- `System` sert à indiquer des systèmes et des appareils génériques

Identifiant de type : `OrderNumber`

`OrderNumber` est l'identifiant de type général pour tous les modules dans le catalogue de matériel, à l'exception de GSD. Les identifiants de type AML ne correspondent pas toujours aux identifiants de type TIA Portal Openness. Les identifiants de type AML n'ont pas d'information sur `FirmwareVersion`. Les informations sur la version de firmware se trouvent dans l'attribut AML séparé "`FirmwareVersion`".

Le format pour cet identifiant de type est l'un des suivants :

- `<OrderNumber>`
Exemple : Numéro de référence : 3RK1 200-0CE00-0AA2

Remarque

Caractères génériques dans les numéros de référence

Il existe quelques rares modules dans le catalogue du matériel, qui utilisent des caractères "génériques" représentant un certain groupe de matériel réel dans leurs numéros de référence, par exemple les différentes longueurs des châssis de S7-300.

Dans ce cas, l'on peut utiliser à la fois les `OrderNumber` spécifiques et les "caractères génériques" `OrderNumber` pour créer une instance de l'objet matériel. Vous ne pouvez toutefois pas utiliser les caractères génériques à n'importe quel emplacement. Exemple : Un châssis de S7-300 peut être créé comme suit :

Numéro de référence : 6ES7 390-1***0-0AA0

ou

Numéro de référence : 6ES7 390-1AE80-0AA0

Veuillez noter que la structure suivante ne peut pas être utilisée par exemple :

Numéro de référence : 6ES7 390-1AE80-0A*0

La valeur retournée de l'identification de type lue est toujours le numéro de référence du catalogue du matériel.

Exemple : Numéro de référence : 6ES7 390-1AE80-0AA0 correspond à Numéro de référence : 6ES7 390-1***0-0AA0

Identifiant de type : GSD

L'identifiant de type pour les appareils basés sur GSD et GSDML est `TypeIdentifier = GSD:<Identifieur>`

L'identifiant se compose des éléments suivants

- `GsdName` : nom du GSD ou GSDML en lettres majuscules
- `GsdType` : est l'un des suivants :
 - D : appareil (Device)
 - R : châssis (Rack)
 - DAP : module de tête
 - M : module
 - SM : sous-module
- `GsdId` : ID du GSD ou GSDML

Les formats suivants pour les identifiants de type sont pris en charge pour l'importation/exportation CAx :

- GSD.<GsdName>/<GsdType>

Exemples :

GSD:SIEM8139.GSD/DAP

GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCP-20140313.XML/D

- <GsdName>/<GsdType>/<GsdId>

Exemple :

GSD:SIEM8139.GSD/M/4

GSD:GSDML-V2.31-SIEMENS-SINAMICS_G110M-20140704.XML/M/IDM_DRIVE_47

Identifiant de type : Système

System. est l'identification des objets qui ne peuvent pas être définis par d'autres identifications. Les formats de ces `TypeIdentifierType` sont les suivants :

- <SystemTypeIdentifier>

Exemples :

System:Device.S7300

System:Subnet.Ethernet

- <SystemTypeIdentifier>/<AdditionalTypeIdentifier>

La `AdditionalTypeIdentifier` est requise si `SystemTypeIdentifier` n'est pas univoque.

La `SystemTypeIdentifier` a un préfixe pour certains types d'objet :

Subnet.

Device.

Rack.

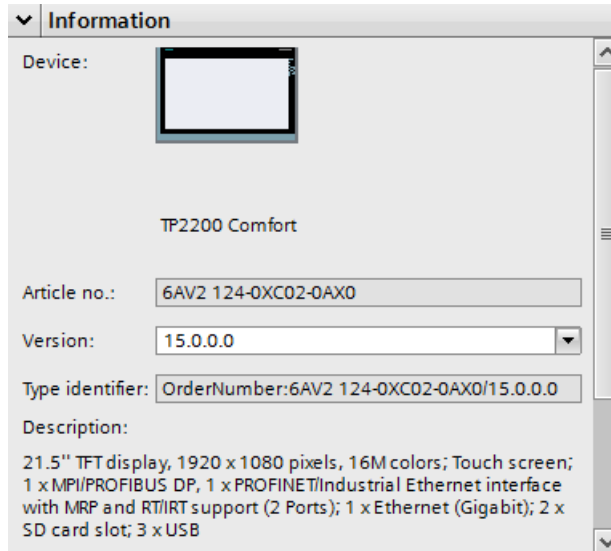
Exemple : System:Rack.S71600/Large

Un châssis avec numéro de référence est identifié via la `OrderNumber`.

Afficher les identifiants de type dans TIA Portal

Lorsque vous devez connaître un identifiant de type, vous le déterminez comme suit dans TIA Portal :

1. Activez le paramètre "Affichage de l'identifiant de type pour les appareils et les modules" sous "Options > Paramètres > Configuration matérielle > Affichage de l'identifiant de type".
2. Ouvrez l'éditeur "Appareils & Réseaux".
3. Sélectionnez un appareil dans le catalogue.
L'identifiant de type s'affiche sous "Information".



8.5.6 Exportation de données CAx

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Dans TIA Portal, vous pouvez exporter votre configuration dans un fichier AML, dans l'éditeur "Appareils et réseaux". Cette fonction basée sur TIA Portal Openness permet d'exporter des données de matériel du niveau projet ou appareils.

TIA Portal Openness offre les moyens suivants pour exporter des données CAx :

- Fonction d'exportation
La fonction d'exportation est accessible par le service `CaxProvider`. Pour le service `CaxProvider`, appelez la méthode `GetService` pour l'objet `Project`.
- Interface de ligne de commande
Vous exécutez "Siemens.Automation.Cax.AmiHost.exe" sous "C:\Programme\Siemens\Automation\Portal V..\Bin\" en transmettant des arguments spécifiques dans la ligne de commande.

Restrictions pour CAx relatives à l'exportation et à l'importation

CAx ne prend pas en charge l'exportation et l'importation

- des connexions port-port
- des connexions vers et entre des châssis d'extension
- des multi CPU
- des appareils H
- des appareils IHM
- du mode de sortie et de la plage de sortie de voies analogiques
- des adresses comprimées

CAx ne prend pas en charge l'exportation et l'importation des appareils et des entraînements suivants :

- 6GK5 414-3FC00-2AA2
- 6GK5 414-3FC10-2AA2
- 6GK5 495-8BA00-8AA2
- 6GK5 496-4MA00-8AA2
- 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
- 6ES7 XXX-XXXXX-XXXX
- 6ES7 370-0AA01-0AA0
- 6ES7 451-3AL00-0AE0
- 6ES7 513-1RL00-0AB0
- 6ES7 515-2RM00-0AB0
- 6ES7 517-3HP00-0AB0

- 6ES7 590-1***0-0AA0//HR
- 6AV2 104-0XXXX-XXXX
- System:Device.S71500/HR
- System:IPIProxy.Device
- System:IPIProxy.Rack

Code de programme : accéder au service `CaxProvider`

Pour accéder au service `CaxProvider`, modifiez le code de programme suivant :

```
Project project = tiaPortal.Projects.Open(...);
CaxProvider caxProvider = project.GetService<CaxProvider>();

if(caxProvider != null)
{
    // Perform CAx export and import operation
}
```

Exportation CAx au niveau projet

Pour exporter des données CAx au niveau projet, utilisez la méthode `Export` avec les paramètres suivants :

Nom	Exemple	Description
ProjectToExport	tiaPortal.Projects[0]	Objet projet à exporter
ExportFilePath	new FileInfo(@"D:\Temp\ProjectExport.aml")	Chemin complet de fichier d'exportation pour le fichier AML
LogFilePath	new FileInfo(@"D:\Temp\ProjectExport_Log.log")	Chemin complet du fichier-journal

Pour exporter toutes les données CAx au niveau du projet, modifiez le code de programme suivant :

```
caxProvider.Export(project, new FileInfo(@"D:\Temp\ProjectExport.aml"),
new FileInfo(@"D:\Temp\ProjectExport_Log.log"));
```

Exportation CAx au niveau appareils

Pour exporter des données CAx au niveau appareils, utilisez la méthode `Export` avec les paramètres suivants :

Nom	Exemple	Description
DeviceToExport	project.Devices[0]	Objet appareil à exporter
ExportFilePath	new FileInfo(@"D:\Temp\ProjectExport.aml")	Chemin complet de fichier d'exportation pour le fichier AML
LogFilePath	new FileInfo(@"D:\Temp\ProjectExport_Log.log")	Chemin complet du fichier-journal

Pour exporter toutes les données CAx au niveau du projet, modifiez le code de programme suivant :

```
caxProvider.Export(device, new FileInfo(@"D:\Temp\DeviceExport.aml"), new
FileInfo(@"D:\Temp\DeviceExport_Log.log"));
```

Exportation CAx via la ligne de commande

Pour exporter des données CAx via la ligne de commande, utilisez `Siemens.Automation.Cax.AmiHost.exe` avec les paramètres suivants :

Paramètre	Exemple	Description
-p	-p "D:\Temp\MyProject.ap14"	Indique un chemin d'accès complet à un projet TIA Portal.
-d	-d "S7300/ET200M station_1"	Paramètre optionnel. Si aucun appareil n'est indiqué, l'exportation est effectuée au niveau projet. Indique le nom de l'appareil ou de la station à exporter au sein du projet TIA spécifié.
-m	-m "AML"	Indique le mode d'importation/exportation (format d'exportation/importation):. "AML" exporte au format AML
-e	-e "D:\Import" -e "D:\Import\CAx_Export.aml"	Indique le chemin complet d'accès au fichier AML à exporter. Le nom du projet est utilisé comme nom du fichier exporté, si seul un chemin d'accès est indiqué.

Pour exporter toutes les données CAx au niveau projet via la ligne de commande, modifiez le code de programme suivant :

```
Siemens.Automation.Cax.AmiHost.exe -p "D:\Temp\MyProject.ap14" -m "AML" -e
"D:\Import\CAx_Export.aml"
```

Pour exporter des données CAx au niveau appareil via la ligne de commande, modifiez le code de programme suivant :

```
Siemens.Automation.Cax.AmiHost.exe -p "D:\Temp\MyProject.ap14" -d "S7300/ET200M_station_1" -m "AML" -e "D:\Import\CAx_Export.aml"
```

8.5.7 Importation de données CAx

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)

Utilisation

Dans TIA Portal, vous pouvez importer votre configuration dans un fichier AML, dans l'éditeur "Appareils et réseaux". Cette fonction permet d'importer des données de matériel du niveau projet ou appareils.

TIA Portal Openness offre les moyens suivants pour exporter des données CAx :

- Fonction d'importation
La fonction d'importation est accessible par le service `CaxProvider`. Pour le service `CaxProvider`, appelez la méthode `GetService` pour l'objet `Project`.
- Ligne de commande
Vous exécutez "Siemens.Automation.Cax.AmiHost.exe" sous "C:\Programme\Siemens\Automation\Portal V..\Bin\" en transmettant des arguments spécifiques dans la ligne de commande :

Code de programme : accéder au service `CaxProvider`

Modifiez le code de programme suivant :

```
//Access the CaxProvider service
Project project = tiaPortal.Projects.Open(...);
CaxProvider caxProvider = project.GetService<CaxProvider>();

if(caxProvider != null)

{
    // Perform Cax export and import operation
}
```

Importation CAx

Pour importer des données CAx dans un projet de TIA Portal, utilisez la méthode `Import` avec les paramètres suivants :

Nom	Exemple	Description
<code>ImportFilePath</code>	<code>new FileInfo(@"D:\Temp\ProjectExport.aml")</code>	Chemin complet de fichier d'importation pour le fichier AML
<code>LogFilePath</code>	<code>new FileInfo(@"D:\Temp\ProjectExport_Log.log")</code>	Chemin complet du fichier-journal
<code>ImportOptions</code>	<code>CaxImportOptions.MoveToParkingLot</code> <code>CaxImportOptions.RetainTiaDevice</code> <code>CaxImportOptions.OverwriteTiaDevice</code>	Stratégies pour résoudre le conflit lors de l'importation dans un projet existant non vide.

Pour importer les données CAx, modifiez le code de programme suivant :

```
caxProvider.Import(new FileInfo(@"D:\Temp\ProjectImport.aml"), new
FileInfo(@"D:\Temp\ProjectImport_Log.log"),
CaxImportOptions.MoveToParkingLot);
```

Les `CaxImportOptions` suivantes sont données :

Option pour importation	Description
<code>MoveToParkingLot</code>	Conserver dans le projet l'appareil/les appareils avec conflit de nom et importer dans un dossier parking ceux provenant des données CAx
<code>RetainTiaDevice</code>	Conserver dans le projet l'appareil/les appareils avec conflit de nom et ne pas importer ceux provenant des données CAx
<code>OverwriteTiaDevice</code>	Écraser l'appareil/les appareils avec conflit de nom dans le projet avec ceux provenant des données CAx

Importation CAx via la ligne de commande

Pour importer des données CAx via la ligne de commande, utilisez `Siemens.Automation.Cax.AmiHost.exe` avec les paramètres suivants :

Paramètre	Exemple	Description
<code>-p</code>	<code>-p "D:\Temp\MyProject.ap14"</code>	Indique un chemin d'accès complet à un projet TIA Portal.
<code>-m</code>	<code>-m "AML"</code>	Indique le mode d'importation/exportation (format d'exportation/importation):. "AML" exporte au format AML
<code>-i</code>	<code>-i "D:\Import\CAx_Export.aml"</code>	Indique le chemin complet d'accès au fichier AML à importer.
<code>-c</code>	<code>-c "ParkingLot"</code>	Indique différentes stratégies en cas de conflits de nom d'appareil dans les options d'importation.

Pour importer des données CAx via la ligne de commande, modifiez le code de programme suivant :

```
Siemens.Automation.Cax.AmiHost.exe -p "D:\Temp\MyProject.ap14" -m "AML" -i "D:\Import\CAx_Export.aml"
```

Les options d'importation suivantes sont disponibles :

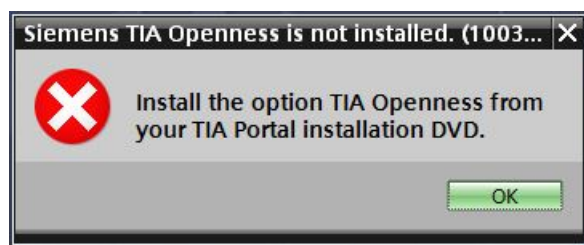
Option pour importation	Description
ParkingLot	Conserver dans le projet l'appareil/les appareils avec conflit de nom et importer dans un dossier parking ceux provenant des données CAx
RetainTia	Conserver dans le projet l'appareil/les appareils avec conflit de nom et ne pas importer ceux provenant des données CAx
OverwriteTia	Écraser l'appareil/les appareils avec conflit de nom dans le projet avec ceux provenant des données CAx

8.5.8 Exception pour l'importation et exportation de données CAx

Exception en raison de la non-disponibilité de TIA Openness

La mise en œuvre de CAx est basée sur des API TIA Openness publiques. Les API Openness publiques ne sont disponibles que si l'utilisateur a installé le pack optionnel Openness au cours de l'installation de TIA Portal. Il est donc important de vérifier si Openness est disponible avant l'exécution de fonctions liées à CAx. (voir Installation de TIA Openness (Page 25))

Lorsque l'utilisateur déclenche une action d'exportation ou d'importation CAx dans l'interface utilisateur de TIA Portal, un contrôle est toujours effectué afin de déterminer si TIA Openness est disponible dans le système. Si aucune installation de TIA Openness n'est trouvée, une boîte de dialogue TIA Portal contenant le message d'erreur suivant est affichée.



Si l'opération CAx est exécutée via la ligne de commande, la boîte de dialogue suivante est affichée en cas de non-disponibilité de TIA Openness.

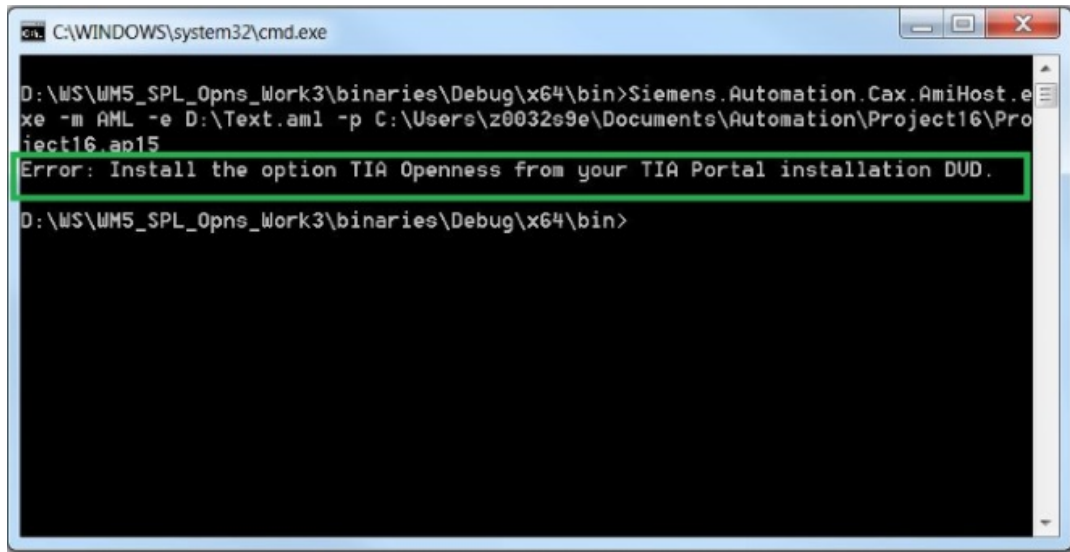


Figure 8-3 OpennessNotInstalled-Commandline

8.5.9 Appareils et modules Round-Trip

Conditions requises

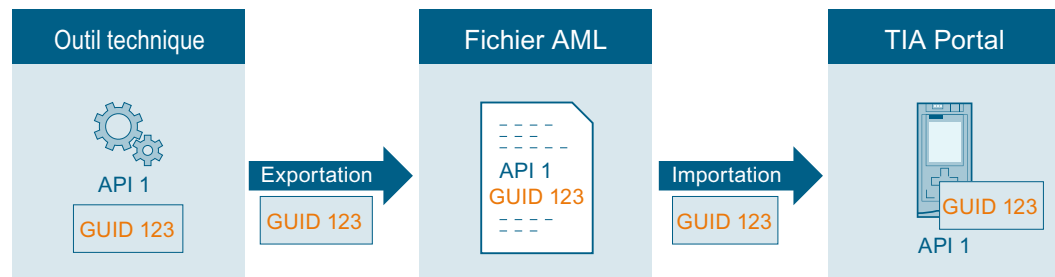
- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

Vous pouvez échanger des données de configuration entre TIA Portal et d'autres outils d'ingénierie tels qu'un outil de planification électrique comme EPLAN ou l'outil de sélection TIA Selection Tool. Pour identifier les appareils à importer et exporter, on utilise un identifiant global unique, AML GUID.

Pendant les processus Round-Trip, l'AML GUID pour les objets physiques, tels que les appareils et les éléments d'appareils qui ne sont pas intégrés (par ex. CPU ou modules), reste stable. Ce qui n'est pas le cas pour les objets virtuels comme les variables, les voies, etc.

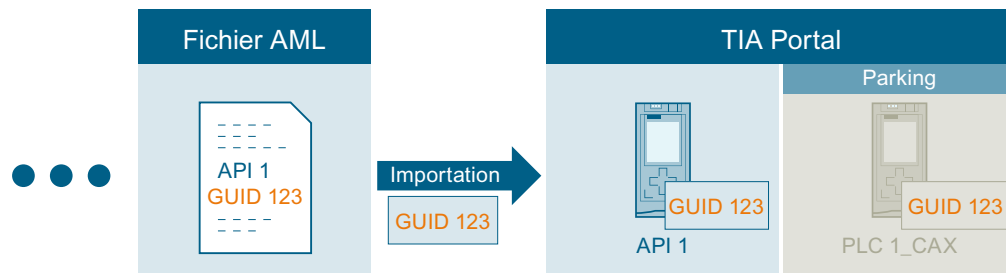
Pendant la première exportation à partir de TIA Portal, l'AML GUID est généré de manière aléatoire pour un appareil ou un élément d'appareil non intégré mais stabilisé ensuite.



Si vous exportez un appareil à partir d'un outil technique dans un projet TIA Portal vide, AML GUID est ajouté dans le commentaire de l'objet matériel. Lorsque le paramètre correspondant est activé dans TIA Portal sous "Outils > Paramètres > CAX > Paramètres d'importation", l'AML GUID est ajouté dans la langue d'édition actuelle. Le processus Round-Trip ne prend en charge qu'une langue d'éditeur pour les AML GUID. Si vous importez ou exportez des données, utilisez toujours la langue d'édition dans laquelle vous avez commencé le Round Trip.

AML GUID reste identique pour cet objet matériel pour toutes les importations et exportations suivantes. Les modifications effectuées sur l'objet matériel sont appliquées.

Les noms d'objets doivent être uniques au sein d'un projet TIA Portal. L'importation d'un appareil ou d'un élément d'appareil dans un projet TIA Portal dans lequel un objet avec le même nom existe déjà entraînerait un conflit de nom. Lors de l'importation, vous avez la possibilité de déplacer les objets avec les conflits de noms dans l'emplacement de stockage personnalisé. "_CAX" est ajouté au nom de l'objet importé.



Remarque

Copie d'un appareil importé

Si vous copiez un appareil ou un élément d'appareil avec un AML GUID, vous devez supprimer l'AML GUID dans le commentaire de l'objet copié. Faute de quoi, il existe des appareils ou des éléments d'appareils ayant des AML GUID identiques dans votre projet, ce qui aboutit à un fichier AML invalide.

Paramétrage de l'importation

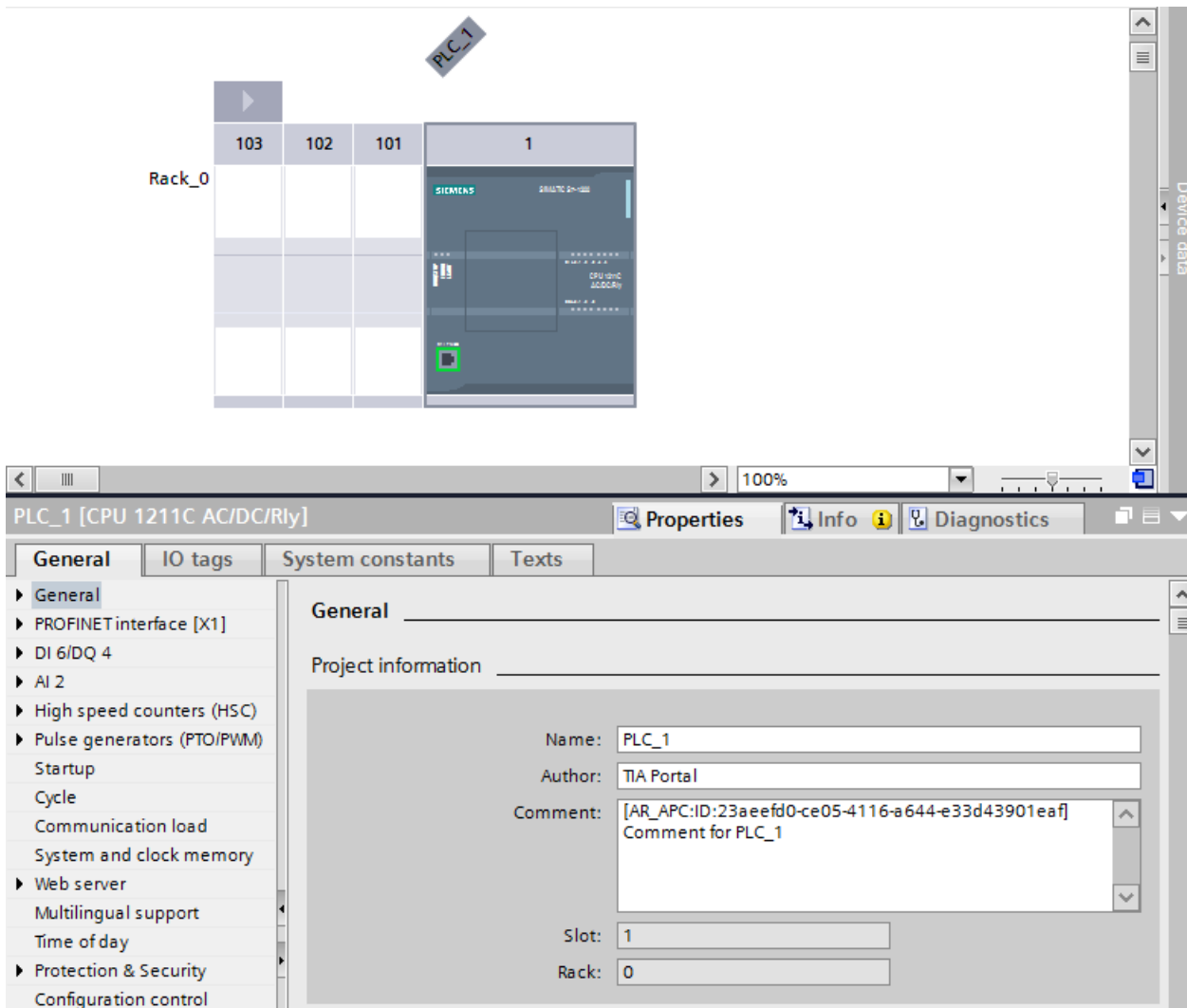
1. Définissez le nom de dossier de l'emplacement de stockage sous "Options > Paramètres > CAx > Paramètres de résolution de conflit".
Le dossier de l'emplacement de stockage sert à enregistrer des objets avec des conflits de noms.
2. Activez "Options > Paramètres > CAx > Paramètres d'importation > Enregistrer GUID pendant l'importation".

Remarque

AML GUID valides

Si vous éditez un AML GUID avant l'importation, celui-ci n'est plus valide et l'importation est annulée.

Après l'importation dans le TIA Portal, l'AML GUID est ajouté dans le commentaire d'utilisation existant comme suit :



Remarque**Commentaire trop long**

Si l'annexe du commentaire AML GUID dépasse la longueur maximale de 500 caractères, le commentaire de l'utilisateur est limité à 500 caractères. Une information correspondante est journalisée.

Structure AML

L'identification créée est exportée dans un fichier AML comme décrit dans la section de code suivante :

```
<InternalElement ID="23aeefd0-ce05-4116-a644-e33d43901eaf"  
Name="PLC_1"
```

8.5.10 Topologie de l'exportation/importation**Conditions requises**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

Dans TIA Portal, vous pouvez exporter les appareils avec leurs informations de topologie dans un fichier AML. En cas d'importation dans un projet TIA Portal vide, les éléments d'appareil importés conservent les informations de topologie.

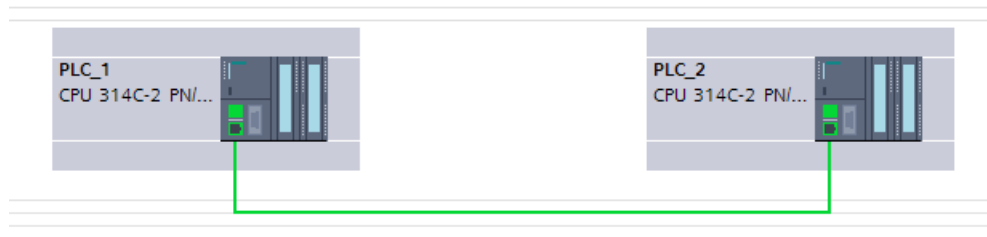
L'élément <InterLink> fournit des informations de liaison de connexions port à port entre les éléments d'appareil. Il apparaît sous l'objet de niveau supérieur commun des appareils connectés et contient des noms de variables uniques.

Attributs d'un élément "InternalLink"

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Manipulation	Commentaire
Name	Obligatoire	Les noms de variables sont formatés en tant que "Link to Port_n" (n signifiant un nombre compris entre 1 et le nombre de connexions port à port).
RefPartnerSideA	Obligatoire	Désigne le port relié. Formaté en tant que UniqueIDOfPort:CommunicationPortInterface
RefPartnerSideB	Obligatoire	Désigne le port relié. Formaté en tant que UniqueIDOfPort:CommunicationPortInterface

Exemple : vue topologique



Structure AML

La figure suivante montre une structure d'élément partielle du fichier AML exporté. Elle contient deux ID uniques pour les ports dans des API.

```

...
<InternalElement ID="e1966b52-b8b3-47b4-8866-a754ebb77648" Name="Port_1">
  <Attribute Name="Label" AttributeDataType="xs:string">
...
<InternalElement ID="75f31daf-575f-48a2-ab35-8f07a376eb1b" Name="Port_1">
  <Attribute Name="Label" AttributeDataType="xs:string">

```

L'élément <InternalLink> contient trois attributs obligatoires.

```

<InternalLink Name="Link to Port_1"
  RefPartnerSideA="e1966b52-b8b3-47b4-8866-a754ebb77648:CommunicationPortInterface"
  RefPartnerSideB="75f31daf-575f-48a2-ab35-8f07a376eb1b:CommunicationPortInterface" />

```

8.5.11 Exportation d'un élément d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

L'objet `Device` est un objet conteneur pour une configuration centralisée ou décentralisée. Il constitue l'objet de niveau supérieur pour les objets `DeviceItem` et l'élément interne de niveau le plus élevé de la hiérarchie d'instance du projet TIA Portal dans la structure d'un fichier AML.

L'exportation de données CAx prend en charge les types d'appareil indiqués par l'identifiant de type AML :

- Modules physiques
- Appareils basés sur GSD/GSDML
- Systèmes

Les appareils peuvent être regroupés dans un objet `DeviceUserFolder`.

Remarque

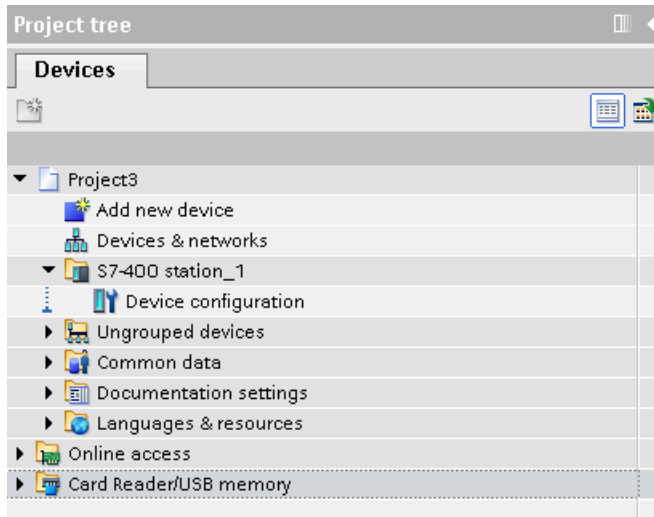
Lors de l'exportation d'un appareil individuel, tous les sous-réseaux du projet sont exportés eux aussi.

Attributs

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Manipulation	Commentaire
Name	Obligatoire	
TypeIdentifier	Obligatoire	
Comment	Optionnel	Paramétrage par défaut : ""

Exemple : Configuration exportée



Structure AML du fichier d'exportation

L'exemple de structure suivant représente l'exportation de l'appareil individuel "S7-400 station_1" sans châssis ni module :

```

...
<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>

```

Voir aussi

Structure des données CAx pour l'importation/exportation (Page 507)

Identifiants de type AML (Page 512)

8.5.12 Importation d'un objet d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

L'objet `Device` est un objet conteneur pour une configuration centralisée ou décentralisée. Il constitue l'objet de niveau supérieur pour les objets `DeviceItem` et l'élément interne de niveau le plus élevé de la hiérarchie d'instance du projet TIA Portal dans la structure d'un fichier AML.

L'importation de données CAx prend en charge les types d'appareil indiqués par l'identifiant de type AML :

- Modules physiques
- Appareils basés sur GSD/GSDML
- Systèmes
- Appareils génériques

Quand il y a déjà un objet `DeviceUserFolder` dans un projet TIA Portal, les appareils sont regroupés dans le dossier correspondant.

Si vous connaissez uniquement l'identifiant (`TypeIdentifier`) d'un module de tête ou d'un API et pas celui du châssis correspondant et de l'appareil, vous pouvez importer un châssis générique.

Exemple : `TypeIdentifier = System:<Prefix>.Generic`

Pour l'échange d'appareils génériques, il faut que les éléments suivants soient présents dans le châssis décrit dans le fichier AML.

- Appareils centralisés : AP
- Appareils décentralisés : module de tête

Si les appareils sont génériques, l'attribut `BuiltIn` définit le type de châssis ou de module :

- Physique : `BuiltIn = True`
- Générique : `BuiltIn = False`

Exemple : importer des appareils génériques

L'exemple de structure suivant représente l'importation de l'appareil générique "S7-400 station" sans châssis ni module.

```

...
<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>

```

Exemple : importer la hiérarchie d'un dossier utilisateur d'un appareil

L'exemple de structure suivant montre l'importation d'une hiérarchie de dossiers.

```

...
<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>
...

```

Hiérarchie importée des dossiers utilisateur

La hiérarchie suivante est importée dans le navigateur de projet :

▼	Group_1	
▼	Group_2	
▼	Group_3	
▼	Group_4	

Voir aussi

Structure des données CAx pour l'importation/exportation (Page 507)

Identifiants de type AML (Page 512)

8.5.13 Exportation/importation d'un appareil avec une adresse définie

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

Dans TIA Portal, vous pouvez exporter les objets d'adresse d'éléments d'appareils IO dans un fichier AML. En cas d'importation dans un projet TIA Portal vide, les éléments d'appareil importés conservent les objets d'adresse dans les éléments d'appareils IO correspondants.

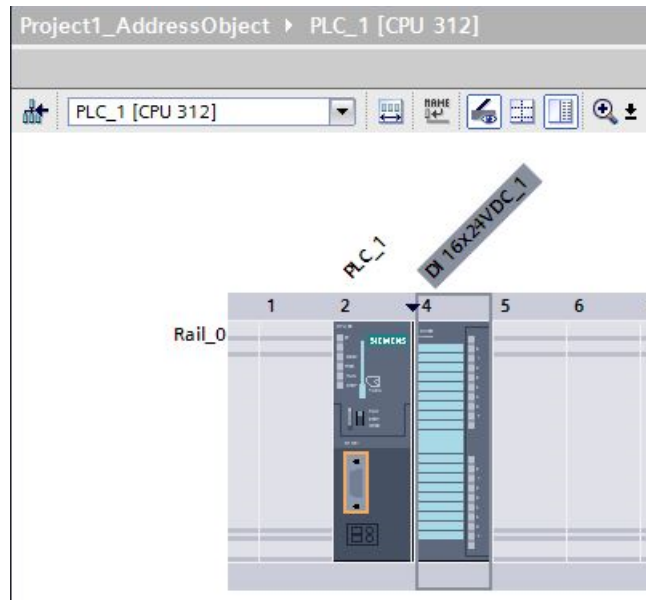
L'attribut Address dans le fichier AML contient RefSemantic avec le paramètre obligatoire sur la valeur indiquée OrderedListType.

Attributs d'un élément "Address"

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Manipulation	Commentaire
IoType	Obligatoire	Entrée ou sortie
Length	Optionnel	Largeur de voie
StartAddress	Obligatoire	Adresse de début de l'appareil IO

Exemple : éléments d'appareils IO avec des objets d'adresse



Structure AML

La figure suivante montre une structure d'élément partielle du fichier AML exporté. Elle contient les éléments Address et leurs attributs.

```
<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 (élagage) décrit le processus d'optimisation du contenu en supprimant certaines choses qui ne doivent pas nécessairement être indiquées dans le fichier XML. Les informations créées automatiquement sur le sous-module ne sont pas indiquées dans ce fichier XML réduit et les objets d'adresse correspondants sont disposés sous l'élément directement supérieur.

La figure suivante montre une structure d'élément partielle du fichier AML exporté avant le 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>
  </Attribute>
</InternalElement>
```

Les informations de sous-module like <InternalElement> sont supprimées du fichier Pruned-AML et les objets d'adresse correspondants y sont conservés.

```
<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>
```

Voir aussi

Pruned AML (Page 503)

8.5.14 Exportation/importation d'un appareil avec des voies

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

Dans TIA Portal, vous pouvez exporter les objets de voie d'éléments d'appareils IO dans un fichier AML. En cas d'importation dans un projet TIA Portal vide, les éléments d'appareil importés conservent les objets de voie dans les éléments d'appareils IO correspondants.

L'élément <ExternalInterface> est représenté dans des éléments internes de nœuds et sous-réseaux et indique que les nœuds et sous-réseaux sont reliés.

Attributs d'un élément "ExternalInterface"

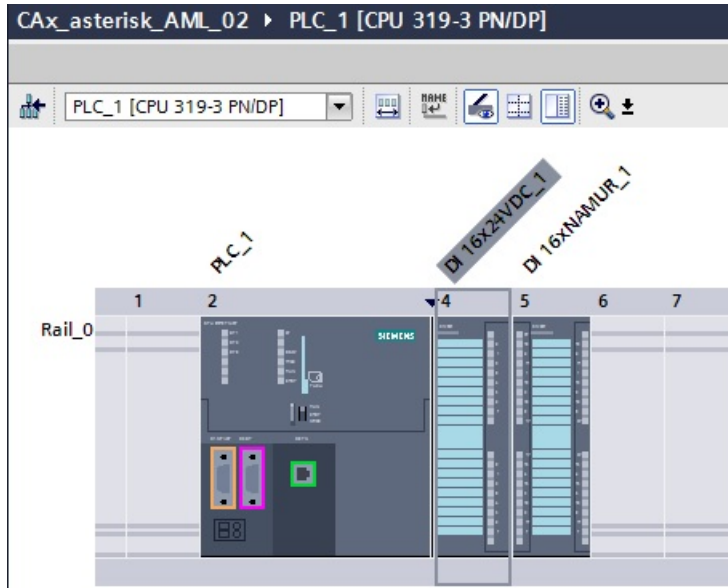
Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Manipulation	Commentaire
IoType	Obligatoire	Entrée ou sortie
Length	Optionnel	Largeur de voie (1 pour les signaux TOR et 16 pour les signaux analogiques)
Number	Obligatoire	Le numéro de voie commence par 0
Type	Obligatoire	Analogique ou TOR

Numérotation de voies

Les voies d'entrées TOR, de sorties TOR, d'entrées analogiques, de sorties analogiques et les voies technologiques sont numérotées comme DI_0, DO_0, AI_0, AO_0, TO_0. Les voies sur les éléments d'appareils eux-mêmes sont numérotées en premier et ensuite les voies sur les éléments d'appareils inférieurs (en profondeur d'abord). Tous les autres éléments d'appareil ont un numéro de voie propre commençant par 0.

Exemple : appareils avec des voies



Structure AML

La figure suivante montre une structure d'élément partielle du fichier AML exporté.

```
<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>
```


8.5.15 Exportation d'objets d'élément d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

L'exportation d'objets d'élément d'appareil n'est utilisable que pour des API.

Les objets `DeviceItem` sont des éléments inférieurs connectés d'un objet `Device` : Un objet du type `DeviceItem` peut être aussi bien un châssis qu'un module enfiché.

- Le premier élément qui suit un appareil est de type "châssis". Le `PositionNumber` d'un châssis commence à 0. S'il existe plusieurs châssis, ceux-ci sont numérotés de manière continue (1, 2, 3, ...).
Il n'existe aucune restriction relative à l'ordre dans un niveau hiérarchique du fichier AML.
- Tous les autres éléments suivants du type "Châssis" sont des modules.

L'exportation de données CAx prend en charge les types d'éléments d'appareil indiqués par l'identifiant de type AML :

- Modules physiques
- Modules GSD/GSDML

Attributs

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

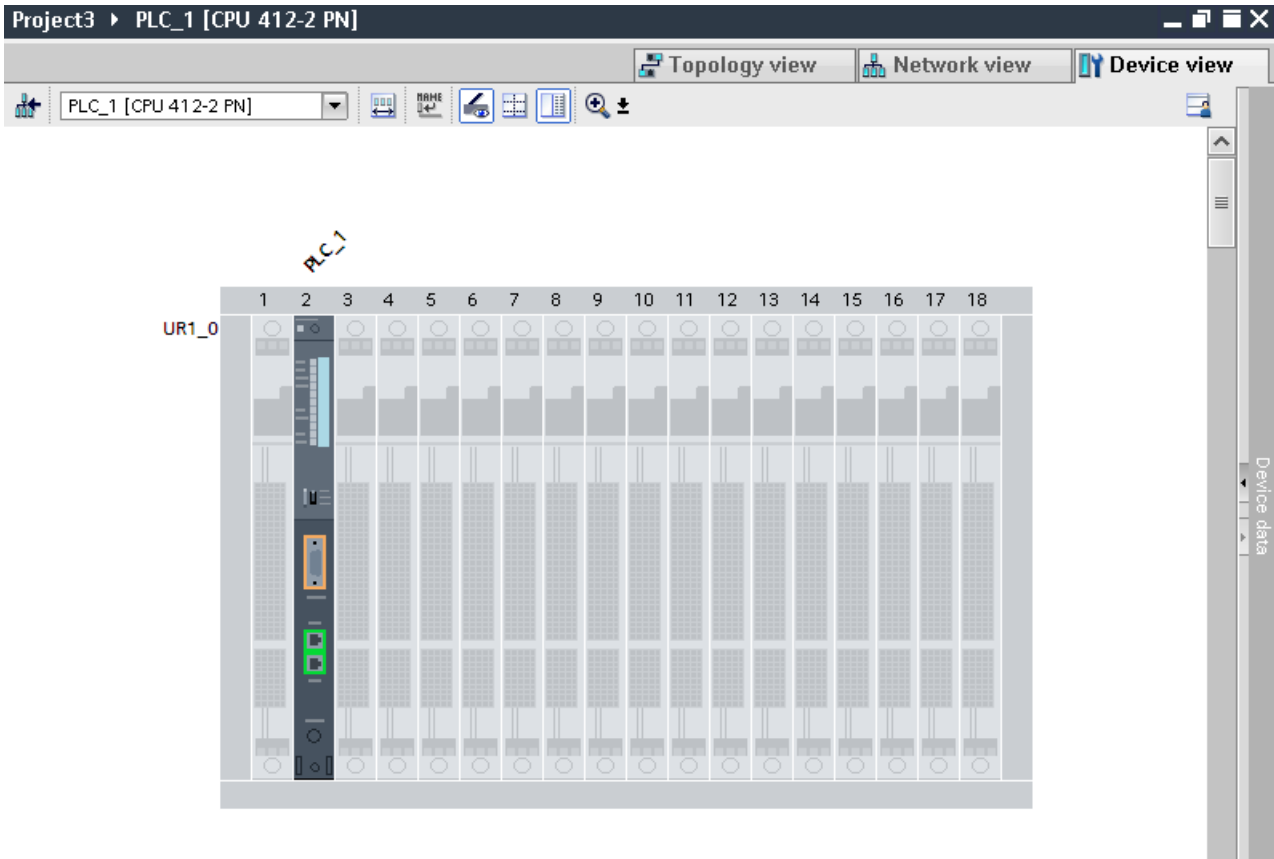
Attribut	Manipulation	Commentaire
Name	Obligatoire Exporter seulement pour "BuiltIn" = TRUE	
TypeName	Exporter seulement pour "BuiltIn" = FALSE	
DeviceItemType	Exporter seulement	Uniquement pour les API (appareils centraux) et éléments d'appareil (châssis physique, modules, module de tête).
PositionNumber	Obligatoire	
BuiltIn	Optionnel	Paramétrage par défaut : FALSE
TypeIdentifier	Obligatoire pour "BuiltIn" = FALSE Ignoré pour "BuiltIn" = TRUE	

Attribut	Manipulation	Commentaire
FirmwareVersion	Optionnel, obligatoire lorsque l'objet prend en charge des versions de firmware	
PlantDesignation IEC	Optionnel	Paramétrage par défaut : ""
LocationIdentifier IEC	Optionnel	Paramétrage par défaut : ""
Comment	Optionnel pour "BuiltIn" = FALSE	Paramétrage par défaut : ""
Address	Optionnel	"Address" ne possède pas d'attributs imbriqués

Le tableau suivant représente les attributs imbriqués des attributs "Address" de l'objet "DeviceItem" :

Attributs pour "Address"	Manipulation	Commentaire
StartAddress	Obligatoire	
Length	Exporter seulement	L'exportation/importation d'adresses de longueur = 0 n'est pas prise en charge.
IoType	Obligatoire	Entrée ou sortie

Exemple : Configuration exportée



Structure AML du fichier d'exportation

L'exemple de structure suivant représente l'exportation de "UR1_0" et du module "PLC_1".

```

...
<InternalElement ID="6563466e-2de9-42ca-951d-eb8f2545958d" Name="S7-400 station_1">
  <Attribute Name="TypeIdentifrier" AttributeDataType="xs:string">
    <Value>System:Device.S7400</Value>
  </Attribute>
<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="TypeIdentifrier" AttributeDataType="xs:string">
    <Value>OrderNumber:6ES7 400-1TA01-0AA0</Value>
  </Attribute>
<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="TypeIdentifrier" 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>
...

```

Voir aussi

Exportation/importation d'appareils et éléments d'appareils basés sur GSD/GSDML (Page 544)

Structure des données CAx pour l'importation/exportation (Page 507)

Identifiants de type AML (Page 512)

8.5.16 Importation d'objets d'élément d'appareil

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Etablissement d'une connexion au portail TIA (Page 74)
- Un projet est ouvert.
Voir Ouvrir un projet (Page 99)
- L'API est hors ligne.

Utilisation

L'importation d'objets d'élément d'appareil n'est utilisable que pour des API.

Les objets `DeviceItem` sont des éléments inférieurs connectés d'un objet `Device` : Un objet du type `DeviceItem` peut être aussi bien un châssis qu'un module enfiché.

- Le premier élément qui suit un appareil est de type `Châssis`. Le `PositionNumber` d'un châssis commence à 0. S'il existe plusieurs châssis, ceux-ci sont numérotés de manière continue (1, 2, 3, ...).
Il n'existe aucune restriction relative à l'ordre dans un niveau hiérarchique du fichier AML.
- Tous les autres éléments suivants du type `Châssis` sont des modules.

L'importation de données CAx prend en charge les types d'éléments d'appareil indiqués par l'identifiant de type AML :

- Modules physiques
- Modules GSD/GSDML
- Modules génériques

Si vous connaissez uniquement l'identifiant (`TypeIdentifier`) d'un module de tête ou d'un API et pas celui du châssis correspondant et de l'appareil, vous pouvez importer un châssis générique.

Exemple : `TypeIdentifier = System:Rack.Generic`

Pour l'échange de châssis génériques, les éléments suivants doivent être présents dans le châssis décrit dans le fichier AML.

- Appareils centralisés : AP
- Appareils décentralisés : module de tête

8.5 Importation/exportation de données matérielles

Un châssis générique provient du type Device. Par conséquent, un fichier AML importé dans TIA Portal peut utiliser l'identifiant de type de ce châssis :

Dans ce cas, c'est TIA Portal qui détermine l'identifiant de type pour le châssis.

Si des châssis et des modules sont génériques, l'attribut `BuiltIn` définit le type de châssis ou de module :

- Physique : `BuiltIn = True`
- Générique : `BuiltIn = False`

Restrictions

Pendant l'importation, l'attribut `DeviceItemType` est sans importance et il est optionnel.

Remarque

Attribut "FirmwareVersion"

Si aucune `FirmwareVersion` n'est indiquée dans le fichier d'importation, la dernière version de firmware disponible dans TIA Portal est utilisée lors de l'importation CAx.

Quand l'attribut `FirmwareVersion` figure dans le fichier d'importation avec une valeur vide, l'importation de l'élément d'appareil échoue et un message d'erreur est consigné dans le journal.

Exemple : importer des appareils génériques

L'exemple de structure suivant représente l'importation du châssis générique "Rack_1".

```

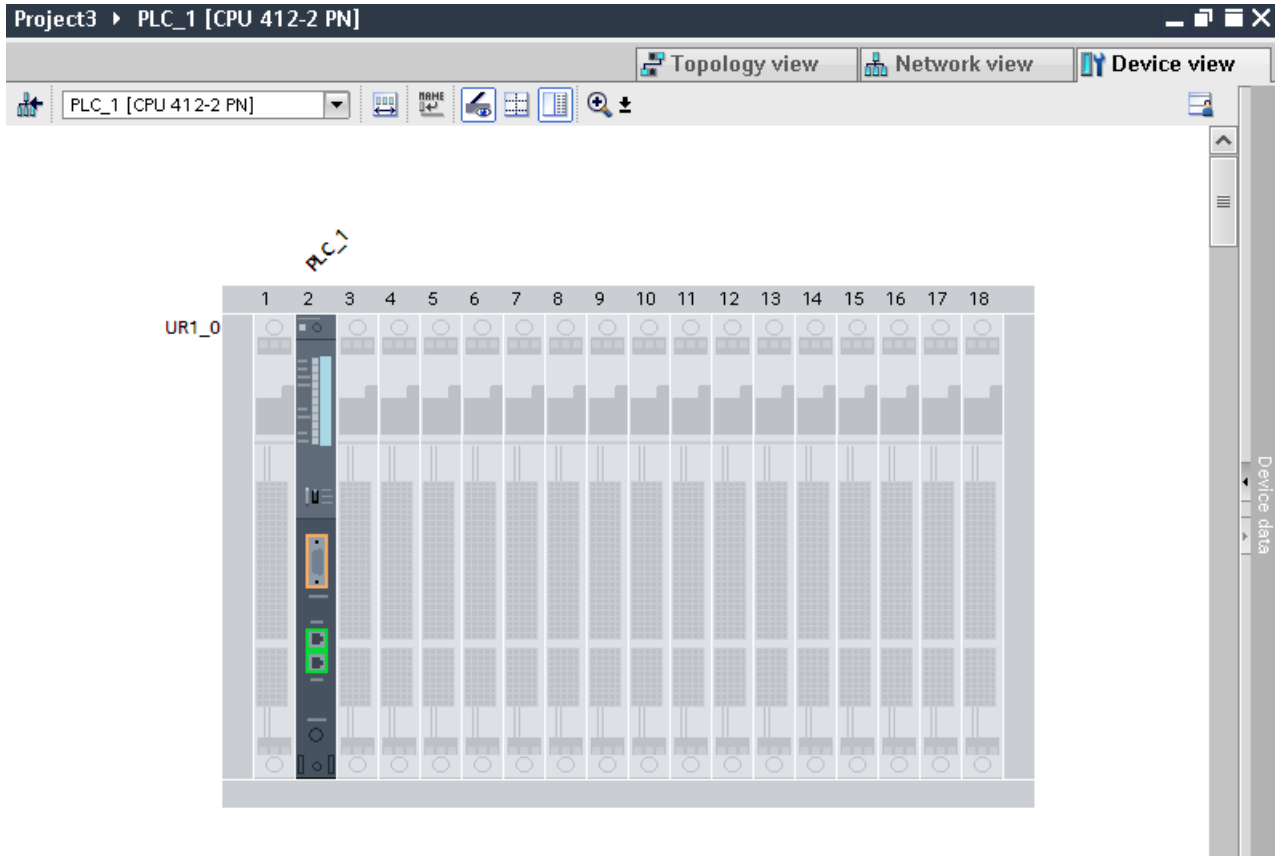
...
<InternalElement ID="6563466e-2de9-42ca-951d-eb8f2545958d" Name="S7-400 station_1">
  <Attribute Name="TypeIdentifiant" AttributeDataType="xs:string">
    <Value>System:Device.S7400</Value>
  </Attribute>
  <InternalElement ID="96930368-14ec-43e2-b9b7-c1fefc4b0534" Name="UR1_0">
    <Attribute Name="TypeName" 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="TypeIdentifiant" AttributeDataType="xs:string">
      <Value>System:Rack.Generic</Value>
    </Attribute>
  <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="TypeIdentifiant" 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>
...

```

Configuration importée

La figure suivante montre la configuration importée dans l'interface utilisateur de TIA Portal :



Voir aussi

Structure des données CAx pour l'importation/exportation (Page 507)

Identifiants de type AML (Page 512)

8.5.17 Exportation/importation d'appareils et éléments d'appareils basés sur GSD/GSDML

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal
- Un projet est ouvert.
Voir Ouverture d'un projet
- L'API est hors ligne.

Utilisation

L'importation/exportation CAx d'appareils et éléments d'appareils basés sur GSD/GSDML est comparable avec l'importation/exportation d'appareils standards.

Les attributs exportables sont différents pour les appareils et éléments d'appareils basés sur GSD/GSDML ; par ex. l'attribut "Label" existe pour GSD/GSDML.

L'importation générique d'appareils et de châssis est possible. Pour l'importation, utilisez la même identification que pour les appareils standard :

- Importation d'appareils génériques : `TypeIdentifieur = System:Device.Generic`
- Importation de châssis génériques : `TypeIdentifieur = System:Rack.Generic`

Si les appareils sont génériques, l'attribut `BuiltIn` définit le type :

- Physique : `BuiltIn = True`
- Générique : `BuiltIn = False`

Attributs pour un appareil

Le tableau suivant montre les attributs correspondants des appareils de fichiers d'importation et d'exportation CAx :

Attribut	Manipulation de l'attribut	Commentaire
Name	Obligatoire pour l'exportation et l'importation	
TypeIdentifieur	Obligatoire pour l'exportation et l'importation	
Comment	Optionnel pour l'importation	

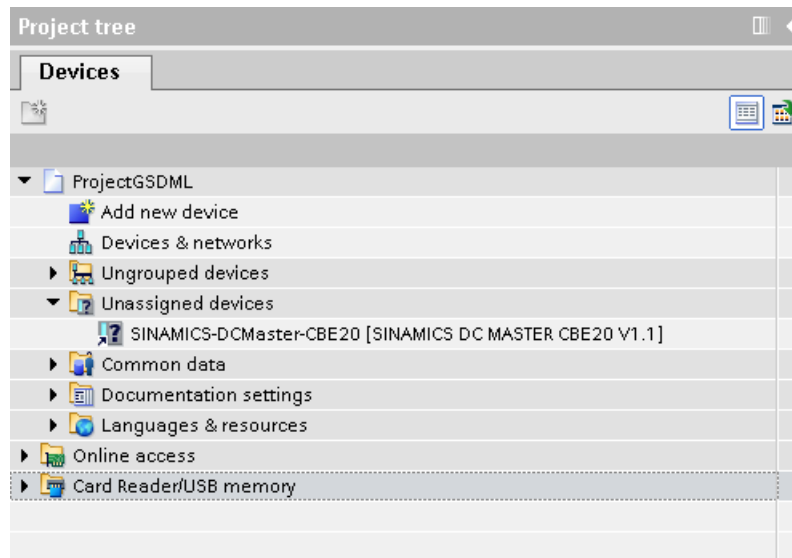
Attributs pour un élément d'appareil

Le tableau suivant montre les attributs correspondants d'un élément d'appareil de fichiers d'importation et d'exportation CAx :

Attribut	Manipulation de l'attribut Intégré = FAUX Éléments d'appareil gé- nériques	Manipulation de l'attribut Intégré = VRAI Éléments d'appareil physiques	Commentaire
Name	Obligatoire	Exporter seulement	
TypeName	Exporter seulement	-/-	
DeviceItem- Type	Exporter seulement	Exporter seulement	Uniquement pour les éléments d'appareil API (appareils centralisés) et les éléments d'appareil module de tête (appareils décentralisés)

8.5 Importation/exportation de données matérielles

Attribut	Manipulation de l'attribut Intégré = FAUX Éléments d'appareil gé- nériques	Manipulation de l'attribut Intégré = VRAI Éléments d'appareil physiques	Commentaire
PositionNum- ber	Obligatoire	Obligatoire pour l'exportation Cas d'exception : Éléments d'appareils de type interfa- ce : Optionnel pour l'importation Éléments d'appareils de type port : Obligatoire pour l'importation d'appa- reils intégrés lorsque l'attribut "La- bel" n'est pas spécifié. Lorsque "Po- sitionNumber" et "Label" sont tous les deux configurés, "PositionNum- ber" reçoit la priorité supérieure pour l'exportation et l'importation.	
BuiltIn	Optionnel		Paramétrage par défaut : FALSE
TypeIdentifieur	Obligatoire pour "BuiltIn" = FALSE	Ignoré pour "BuiltIn" = TRUE	
Comment	Optionnel	-/-	
Label	-	- Éléments d'appareils de type interfa- ce : Obligatoire Éléments d'appareils de type port : Obligatoire lorsque l'attribut "Numé- ro de position" n'est pas spécifié. Lorsque "Numéro de position" et "Marquage" sont tous les deux confi- gurés, "Numéro de position" reçoit la plus grande priorité ; le même princi- pe s'applique pour l'importation.	

Exemple : Appareil GSD/GSDML exporté

Structure AML du fichier d'exportation

La figure suivante indique la structure du fichier AML exporté.

```

...
<InternalElement ID="9ae02cde-dfb4-4d43-a649-68b9ede7fc3d" Name="Ungrouped devices">
  <InternalElement ID="12d4ce0f-346d-4bfa-b139-c9d0db64c794" Name="GSD device_1">
    <Attribute Name="TypeIdentifrier" 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="TypeIdentifrier" 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="TypeIdentifrier" 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>
      ...
    </InternalElement>
  ...

```

Voir aussi

Identifiants de type AML (Page 512)

8.5.18 Exportation/importation de sous-réseaux**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal
- Un projet est ouvert.
Voir Ouverture d'un projet
- L'API est hors ligne.

Structure AML

Les sous-réseaux décrivent un réseau physique, notamment les appareils connectés au même réseau de type PROFIBUS, PROFINET, MPI ou ASI.

Les liaisons entre un réseau et les éléments d'appareil sont représentées comme référence pour l'objet réseau. Il n'y a pas de référence des objets réseau aux éléments d'appareil reliés. Les paramètres de réseau sont enregistrés dans l'objet réseau. Les paramètres concernant une interface de réseau d'un certain élément d'appareil qui est relié à un réseau sont enregistrés dans un objet nœud de réseau dans cet élément d'appareil. La communication est souvent régulée par "Voies", "Ports" et "Interfaces".

Les sous-réseaux sont exportés dans le fichier AML comme éléments internes, avec la classification de rôle "Sous-réseau" dans la hiérarchie d'instances.

Un sous-réseau a les éléments reliés suivants dans la structure AML :

- Élément interne avec la classification de rôle "Nœud"
Détermine l'interface sur un élément d'appareil.
- <InternalLink>
Détermine les partenaires reliés du sous-réseau. Le nom de variable <InternalLink> est unique et est toujours ajouté dans le fichier AML sous l'élément interne du projet.

```

....
<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>
Constata dans des éléments internes de nœuds et sous-réseaux que les nœuds et sous-réseaux sont reliés. Lorsque les nœuds ou sous-réseaux ne sont pas reliés, les éléments <ExternalInterface> n'existent pas pour les nœuds et sous-réseaux.

```

...
<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>
...

```

Utilisation

L'importation/exportation assistée par ordinateur prend en charge les types suivants de sous-réseaux :

- Ethernet
- PROFIBUS
- MPI
- ASi

Attributs d'un élément "sous-réseau"

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Maniement	Commentaire
Name	Obligatoire	
Type	Obligatoire	Ethernet ou PROFIBUS ou MPI ou ASI

Attributs d'éléments "CommunicationInterface"

Le tableau suivant montre les attributs correspondants des objets pour fichiers d'importation et d'exportation CAX :

Attribut	Traitement	Commentaire
Name	Obligatoire	Sans signification pour les éléments d'appareils "fixes".
Label	Obligatoire	L'étiquette manque éventuellement quand "BuiltIn" = TRUE et "PositionNumber" sont indiqués pour l'objet "DeviceItem" correspondant.
TypIdentifier	Obligatoire	
FirmwareVersion	Obligatoire	
TypeName	Uniquement exportation	Sans signification pour les éléments d'appareils "intégrés".
DeviceItem Type	Uniquement exportation	Uniquement pour CPU et module de tête
PositionNumber	Obligatoire	Sans signification pour l'importation d'éléments d'appareils "intégrés".
BuiltIn	Obligatoire pour l'exportation Optionnel pour l'importation	Sans signification pour l'importation d'éléments d'appareils "non intégrés". "False" par défaut pour l'importation.
Comment	Optionnel	Non approprié pour éléments d'appareils "intégrés".

Attributs d'éléments "CommunicationPort"

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Maniement	Commentaire
Name	Obligatoire	Sans signification pour les éléments d'appareils "intégrés".
Label	Obligatoire	L'étiquette peut manquer lorsque "BuiltIn" = TRUE et "PositionNumber" sont déterminés pour l'objet "DeviceItem" correspondant.
TypIdentifier	Obligatoire	
FirmwareVersion	Obligatoire	
TypeName	Uniquement exportation	Sans signification pour les éléments d'appareils "intégrés".

Attribut	Maniement	Commentaire
DeviceItemtype	Uniquement exportation	Uniquement pour CPU et module de tête
PositionNumber	Obligatoire	Uniquement significatif pour l'importation d'éléments d'appareils "intégrés" quand l'attribut "Label" n'est pas indiqué. Quand "PositionNumber" et "Label" sont tous deux configurés, c'est "PositionNumber" qui a la priorité.
BuiltIn	Obligatoire pour l'exportation Optionnel pour l'importation	Sans signification pour l'importation d'éléments d'appareils "non intégrés". "False" par défaut pour l'importation.
Comment	Optionnel	Non approprié pour éléments d'appareils "intégrés".

Attributs d'un élément "nœud"

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

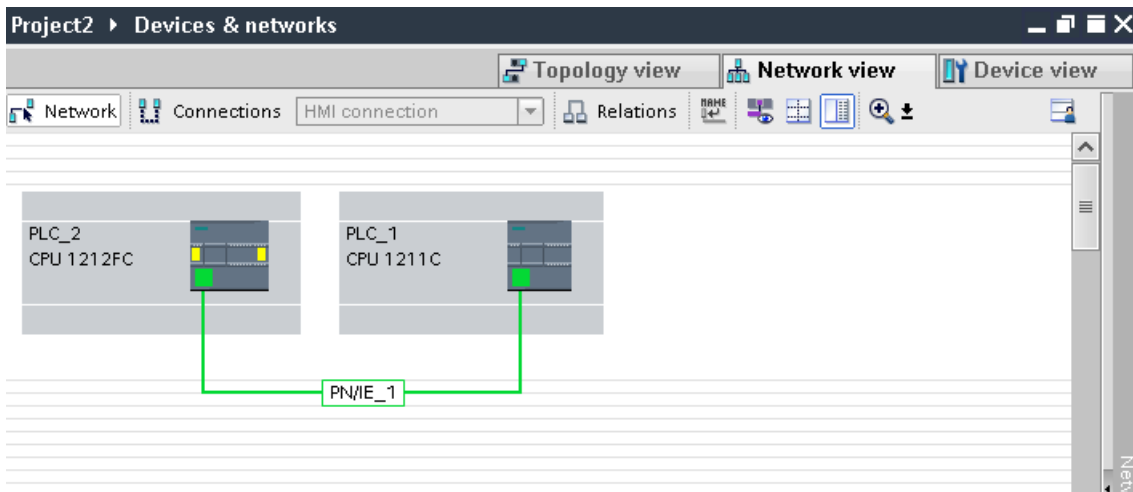
Attribut	Maniement	Commentaire
Name	Uniquement exportation	MPI, PROFIBUS, PROFINET
Type	Exporter seulement	Ethernet ou PROFIBUS ou MPI ou ASi
NetworkAddress	Obligatoire	
SubnetMask	Optionnel	PROFINET La valeur par défaut est conservée pour l'importation quand aucune valeur n'est déterminée.
RouterAddress	Optionnel	PROFINET La valeur par défaut est conservée pour l'importation quand aucune valeur n'est déterminée.
DhcpClientId	Optionnel	PROFINET La valeur par défaut est conservée pour l'importation quand aucune valeur n'est déterminée.
IpProtocolSelection	Optionnel	PROFINET La valeur par défaut est conservée pour l'importation quand aucune valeur n'est déterminée. Valeurs : Project, Dhcp, UserProgram, OtherPath

Attributs d'un élément "voie"

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Maniement	Commentaire
Type	Obligatoire	TOR ou analogique
IoType	Obligatoire	Entrée ou sortie
Number	Obligatoire	
Length	Exporter seulement	

Exemple : sous-réseau exporté



Structure AML

Les figures suivantes montrent la structure du fichier AML exporté.

```

...
<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>
...

```

```

...
<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-b846-30dad8726d1"
        Name="CommunicationPortInterface"
        RefBaseClassPath=
          "AutomationProjectConfigurationInterfaceClassLib/CommunicationPortInterface" />
      <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/CommunicationPort" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
    </InternalElement>
  </InternalElement>
...

```

Voir aussi

- Structure des données CAx pour l'importation/exportation (Page 507)
- Etablissement d'une connexion au portail TIA (Page 74)
- Ouvrir un projet (Page 99)

8.5.19 Exportation/importation de variables API

Conditions

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal
- Un projet est ouvert.
Voir Ouverture d'un projet
- L'API est hors ligne.

Utilisation

Les mnémoniques et les variables exportés et importés sont affectés à un élément d'appareil. L'importation/exportation assistée par ordinateur concerne les mnémoniques et les variables en rapport avec le matériel. Les mnémoniques et les variables sont exportés seulement avec l'élément d'appareil de la consigne de commande, par exemple de la CPU, et non avec d'autres éléments d'appareil auxquels ils se réfèrent, un module d'E/S par exemple. Comme les appareils, les variables sont souvent subdivisées en tables de variables et dans une structure hiérarchique en dossiers.

Éléments de structure AML

Les variables API, tables des variables et dossiers utilisateur de variables peuvent être exportés et importés par la fonction importation/exportation assistée par ordinateur. Les objets variable sont reproduits dans les éléments de structure AML suivants :

- `<InternalElement>`
Les tables de variables et les dossiers utilisateur de variables sont reproduits comme éléments internes de l'API correspondant avec la classification de rôle respective.
- `<ExternalInterface>`
Représente une variable API appartenant à l'élément interne de la table des variables correspondante ou au dossier utilisateur de variables.

Une voie d'affectation avec une variable API est exportée au moyen de l'élément `<internal link>` en tant que partenaire de communication. La structure XML suivante donne un exemple :

```
...
  <InternalLink Name="Link To Tag_1"
    RefPartnerSideA="b33451f6-d88f-4900-8dbe-41f1be1e3535:Channel_DI_0"
    RefPartnerSideB="b2b937ee-d5db-4826-9340-027b1da22828:Tag_1" />
...
```

Dossier utilisateur de variables API

Les objets "TagUserFolder" requièrent seulement l'attribut "Name" dans les fichiers d'importation et exportation assistés par ordinateur.

Attributs d'une table des variables API

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Maniement	Commentaire
Name	Obligatoire, n'est pas pris en considération lorsque "AssignToDefault" = TRUE :	
AssignToDefault	Importer seulement	Utilisé pour identifier la table des variables standard pendant l'importation. Lorsque "AssignToDefault" = TRUE, toutes les variables sont créées dans le cadre de la table des variables standard de TIA Portal.

Attributs d'une variable API

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Maniement	Commentaire
Name	Obligatoire	
DataType	Obligatoire	
LogicalAddress	Obligatoire	Est importé et exporté au format abréviations internationales
Comment	Optionnel	

Exemple : structure AML

La figure ci-dessous montre la structure des objets variable exportés suivants :

- Table de variables standard vide
- Dossier utilisateur de variables "Groupe_1"
- Table de variables incluse "Table de variables_1"
- Quatre variables

```

...
<InternalElement ID="a310ba93-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="f9269ce4-c015-459f-9f59-8f94bca3b186" Name="Tag table_1">
    <ExternalInterface ID="fc0c8c5a-fd5b-443b-b430-6435b6aa22ff" Name="Tag_1"
      RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Tag">
      ....
    </ExternalInterface>
    <ExternalInterface ID="450d6a1d-81b8-49ae-a104-c0072933d669" Name="Tag_2"
      RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Tag">
      ....
    </ExternalInterface>
    <ExternalInterface ID="3de17a36-b5c5-4fc7-9fc3-47e4a8f95087" Name="Tag_3"
      RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Tag">
      <Attribute Name="DataType" AttributeDataType="xs:string">
        <Value>Word</Value>
      </Attribute>
      <Attribute Name="LogicalAddress" AttributeDataType="xs:string">
        <Value>IWO</Value>
      </Attribute>
    </ExternalInterface>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/TagTable" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
    "AutomationProjectConfigurationRoleClassLib/TagUserFolder" />
</InternalElement>
...

```

Voir aussi

Structure des données CAx pour l'importation/exportation (Page 507)

Etablissement d'une connexion au portail TIA (Page 74)

Ouvrir un projet (Page 99)

8.5.20 Exportation/importation de réseaux IO**Conditions**

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal
- Un projet est ouvert.
Voir Ouverture d'un projet
- L'API est hors ligne.

Structure AML

Les réseaux IO sont représentés comme <InternalElement> dans la structure AML.

Les réseaux IO d'un maître ou d'un contrôleur IO sont ajoutés sous l'élément <CommunicationInterface> de l'élément d'appareil d'une interface.

```

...
<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>

```

Les réseaux IO reliés comme esclave ou comme périphérique IO sont ajoutés comme éléments <ExternalInterface> sous l'élément <CommunicationInterface> de l'élément d'appareil d'une interface.

```

<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>

```

Les partenaires reliés du réseau IO sont représentés comme éléments <InternalLink>. Les variables <InternalLink> sont ajoutées sous l'élément supérieur commun d'un réseau IO et de l'élément d'appareil esclave relié, par ex. sous Project, DeviceFolder, DeviceItem.

Le nom de variable <InternalLink> est unique dans l'élément supérieur commun.

Attributs d'un élément "Réseau IO"

Le tableau suivant montre les attributs correspondants des objets de fichiers d'importation et d'exportation assistés par ordinateur :

Attribut	Maniement	Commentaire
Name	Obligatoire	Nom du réseau IO. Quand une chaîne de caractères vide est importée, le réseau IO est créé avec le nom par défaut.
Number	Optionnel	En l'absence d'indication pour l'importation, c'est la valeur par défaut qui est utilisée.

8.5.21 Exportation/importation de commentaires multilingues

Conditions requises

- L'application TIA Portal Openness est connectée à TIA Portal.
Voir Établir une liaison à TIA Portal
- Un projet est ouvert.
Voir Ouverture d'un projet
- L'API est hors ligne.

Utilisation

Commentaires d'exportation et d'importation assistés par ordinateur et commentaires multilingues sur l'échange de données des objets matériels suivants :

- appareils (appareil)
- modules (éléments d'appareils)
- variables (variable)

L'importation/exportation de commentaires multilingues comprend toutes les langues de TIA Portal.

Restrictions

- Exportation
 - Un attribut "Comment" est exporté vers le fichier AML seulement s'il existe un commentaire.
- Importation
 - L'attribut ""Comment" " est facultatif.
 - Pour les éléments d'appareil virtuels, aucun commentaire ne peut être importé.

Exemple : Configuration exportée avec commentaires multilingues

La figure suivante représente la configuration d'un SIMATIC S7 1500 (appareil) avec API_1 (éléments d'appareil). Des commentaires en anglais, français, allemand et chinois sont définis pour les deux objets.

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...

Structure AML

Après l'exportation de ces configurations, les commentaires multilingues sont générés comme des attributs imbriqués de l'appareil, de l'élément d'appareil ou des variables.

- L'attribut de niveau supérieur "Comment" doit posséder la valeur utilisée dans la langue par défaut.
- L'attribut qui suit existe pour chaque commentaire en langue étrangère.

```

...
<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>
...

```

Voir aussi

Structure des données CAx pour l'importation/exportation (Page 507)

Etablissement d'une connexion au portail TIA (Page 74)

Ouvrir un projet (Page 99)

8.5.22 Attributs AML comparés aux attributs TIA Portal Openness

Accéder aux attributs et aux attributs d'exportation/importation

Avec TIA Portal Openness, vous pouvez accéder aux attributs matériels d'objets matériels. Quelques noms que vous utilisez pour accéder à ces attributs, par ex. celui d'un élément d'appareil, diffèrent des noms d'attribut dans le fichier AML d'exportation/importation.

Liste des attributs

Le tableau ci-dessous donne une vue d'ensemble des deux types d'attributs :

Tableau 8-6 Noms d'attribut des appareils et des appareils GSD/GSDML

Fichier AML	TIA Portal Openness
Name	Name
TypenIdentifizier	TypenIdentifizier
Comment	Comment

Tableau 8-7 Noms d'attribut des éléments d'appareil

Fichier AML	TIA Portal Openness
Name	Name
TypenIdentifizier	Affecté au sous-string de <TypenIdentifizier> (c.-à-d. valeur devant premier opérateur "/"), l'élément avec la version de firmware n'étant pas pris en considération. L'affectation du sous-string est valable seulement quand l'identifiant de type commence par le préfixe <OrderNumber:> et comporte un élément avec la version de firmware, autrement l'affectation s'effectue comme complément de <TypenIdentifizier>.
FirmwareVersion	<FirmwareVersion> est affecté au sous-string de <TypenIdentifizier> (c.-à-d. valeur derrière premier opérateur "/"). L'affectation du sous-string est valable seulement quand <TypenIdentifizier> commence par le préfixe <OrderNumber:> et comporte un élément avec la version de firmware.
TypeName	TypeName
DeviceItemtype (pour CPU et module de tête)	Classification
PositionNumber	PositionNumber
BuiltIn	IsBuiltIn
PlantDesignation IEC	PlantDesignation
LocationIdentifier IEC	LocationIdentifier
Comment	Comment

Tableau 8-8 Noms d'attribut des éléments d'appareil GSD/GSDML

Fichier AML	TIA Portal Openness
Name	Name
TypeIdentifier	TypeIdentifier
TypeName	TypeName
DeviceItem Type (pour module de tête)	Classification
PositionNumber	PositionNumber
BuiltIn	IsBuiltIn
Comment	Comment
Label	Label

Tableau 8-9 Noms d'attribut des variables

Fichier AML	TIA Portal Openness
Name	Name
DataType	DataTypeName
LogicalAddress	LogicalAddress
Comment	Comment

Tableau 8-10 Noms d'attribut des tables de variables

Fichier AML	TIA Portal Openness
Name	Name
AssignToDefault	IsDefault

Tableau 8-11 Noms d'attribut des adresses

Fichier AML	TIA Portal Openness
StartAddress	StartAddress
Length	Length
IoType	IoType

Tableau 8-12 Noms d'attribut des ports

Fichier AML	TIA Portal Openness
Name	Name
TypeIdentifier	TypeIdentifier
FirmwareVersion	FirmwareVersion
TypeName	TypeName
PositionNumber	PositionNumber
BuiltIn	IsBuiltIn

Fichier AML	TIA Portal Openness
Comment	Comment
Label	Label

Tableau 8-13 Noms d'attribut des appareils à interface IO

Fichier AML	TIA Portal Openness
Name	Name
TypIdentifier	TypIdentifier
FirmwareVersion	FirmwareVersion
TypeName	TypeName
DeviceItem Type (pour CPU et module de tête)	Classification
PositionNumber	PositionNumber
BuiltIn	IsBuiltIn
Label	Label
Comment	Comment

Tableau 8-14 Noms d'attribut des voies

Fichier AML	TIA Portal Openness
Type	Type
IoType	IoType
Number	Affecté à aucun attribut dans TIA Portal Openness.
Length	ChannelWidth

Principales modifications

9.1 Modifications importantes dans V14 SP1

9.1.1 Modifications importantes dans V14 SP1

Introduction

Les modifications suivantes ont été apportées au modèle d'objet TIA Portal Openness API V14 SP1, ce qui peut avoir des effets sur vos applications existantes :

Modification	Adaptation nécessaire du code de programme
Maniement amélioré des copies maîtres	<p>L'action CreateFrom crée un nouvel objet sur la base d'une copie maître dans une bibliothèque et le place dans la composition là où l'action a été appelée. L'action CreateFrom ne prend en charge que les copies maîtres qui contiennent exclusivement des objets isolés. Le type fourni correspond au type composé respectif.</p> <p>Les compositions suivantes prennent en charge CreateFrom :</p> <ul style="list-style-type: none"> • 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.TechnologicalInstanceDBCComposition • 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

9.1 Modifications importantes dans V14 SP1

Modification	Adaptation nécessaire du code de programme
Maniement amélioré des bibliothèques globales	À présent, les actions existantes avec des bibliothèques globales peuvent englober des actions de modification, par ex. supprimer une copie maître d'une bibliothèque globale. UpdateProject et UpdateLibrary n'utilisent plus les paramètres UpdatePathsMode et DeleteUnusedVersionsMode. Les versions non utilisées ne sont pas effacées après une mise à jour.
Modification de System.String dans System.IO.FileInfo Modification de System.String dans System.IO.DirectoryInfo	Tous les événements pour lesquels il faut spécifier un chemin en chaîne de caractères utilisent le chemin FileInfo ou le chemin DirectoryInfo. Exemple : <ul style="list-style-type: none"> • Ouvrir un projet • Ouvrir une bibliothèque • Créer un projet • Créer une bibliothèque globale • ...

Nouveaux éléments dans le modèle d'objet

Nom	Type	Espace nom	Commentaire
PlcUserConstant	Classe	Siemens.Engineering.SW.Tags	Parties de PlcConstant
PlcUserConstantComposition	Classe	Siemens.Engineering.SW.Tags	Parties de PlcConstantComposition
PlcSystemConstant	Classe	Siemens.Engineering.SW.Tags	Parties de PlcConstant
PlcSystemConstantComposition	Classe	Siemens.Engineering.SW.Tags	Parties de PlcConstantComposition
MultilingualTextItem	Classe	Siemens.Engineering	Accéder aux textes multilingues
MultilingualTextItemComposition	Classe	Siemens.Engineering	Accéder aux textes multilingues
TiaPortalTrustAuthority.FeatureTokens	Valeur d'énumération	Siemens.Engineering	Accéder aux paramètres de TIA Portal
TiaPortalSetting	Classe	Siemens.Engineering.Settings	Accéder aux paramètres de TIA Portal
TiaPortalSettingComposition	Classe	Siemens.Engineering.Settings	Accéder aux paramètres de TIA Portal
TiaPortalSettingsFolder	Classe	Siemens.Engineering.Settings	Accéder aux paramètres de TIA Portal
TiaPortalSettingsFolderComposition	Classe	Siemens.Engineering.Settings	Accéder aux paramètres de TIA Portal
LanguageAssociation	Classe	Siemens.Engineering	Accéder aux langues actives
LanguageComposition.Find	Méthode	Siemens.Engineering	Accéder aux langues actives

Éléments modifiés dans le modèle d'objet

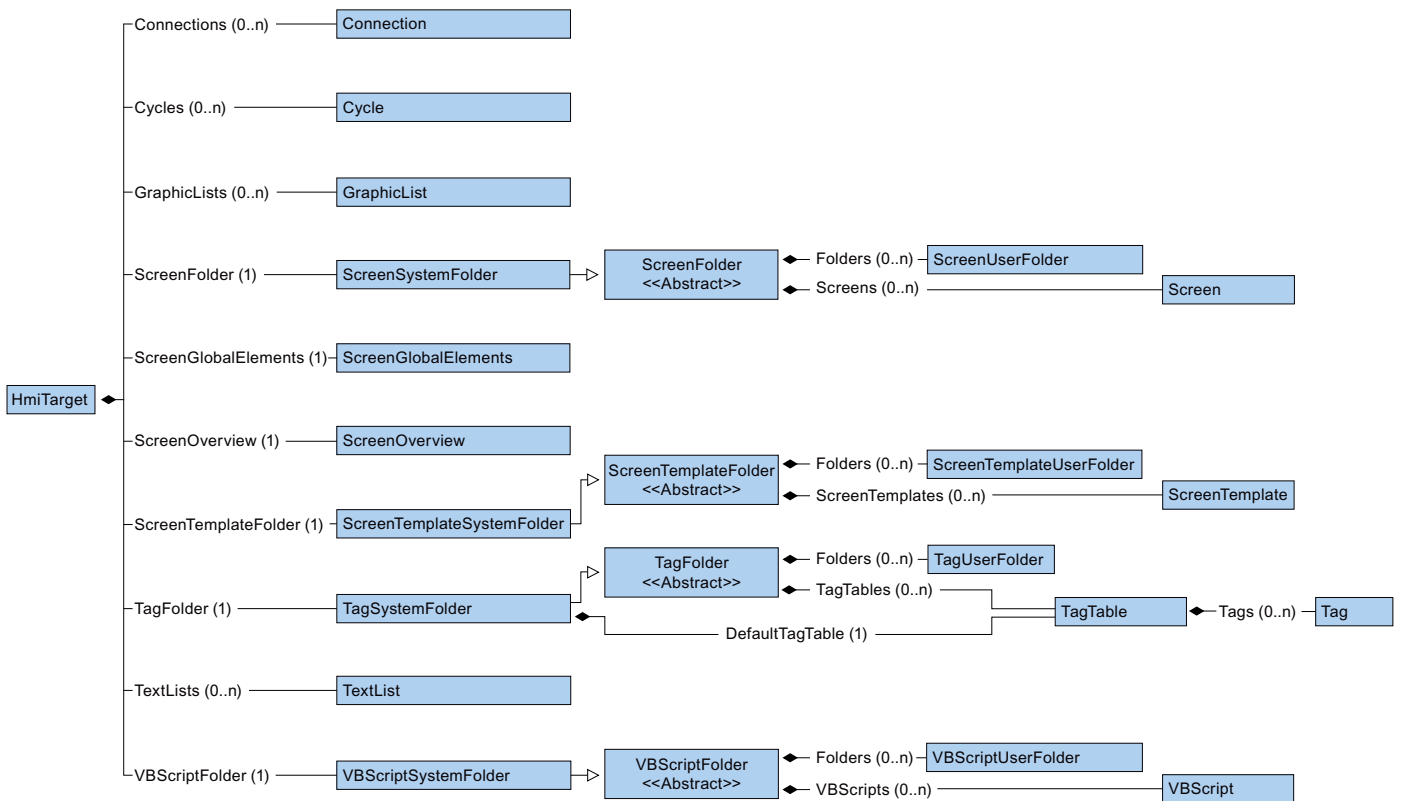
Nom	Type	Espace nom	Commentaire
PlcConstant	Classe	Siemens.Engineering.SW.Tags	Classe de base sortie de PlcUserConstant et de PlcSystemConstant
PlcTag	Classe	Siemens.Engineering.SW.Tags	Parties de PlcConstantComposition
ITargetComparable	Interface	Siemens.Engineering.Compare	Attribut string DataTypeName à la place d'un lien ouvert DataType
MultilingualText	Classe	Siemens.Engineering	Accéder aux textes multilingues
ProjectComposition.Create	Méthode	Siemens.Engineering	Paramètre modifié pour utiliser DirectoryInfo et un string
Project.Subnets	Attribut	Siemens.Engineering	Accéder aux sous-réseaux
Project.Languages	Attribut	Siemens.Engineering	Déplacé pour représenter l'attribut de Siemens.Engineering.LanguageSettings et mettre à disposition les langues prises en charge

Éléments supprimés dans le modèle d'objet

Nom	Type	Espace nom	Commentaire
PlcConstantComposition	Classe	Siemens.Engineering.SW.Tags	Parties dans PlcSystemConstantComposition et PlcUserConstantComposition
CompareResultElement.PathInformation	Attribut	Siemens.Engineering.SW.Tags	N'est plus utilisé
MultilingualText.GetText(CultureInfo cultureInfo)	Méthode	Siemens.Engineering.Compare	Concept modifié pour l'accès aux éléments de texte de MultilingualText
TiaPortalTrustAuthority.CustomerIdentification	Valeur d'énumération	Siemens.Engineering	N'est plus utilisé
TiaPortalTrustAuthority.ElevatedAccessExtensions	Valeur d'énumération	Siemens.Engineering	N'est plus utilisé

Modifications du comportement

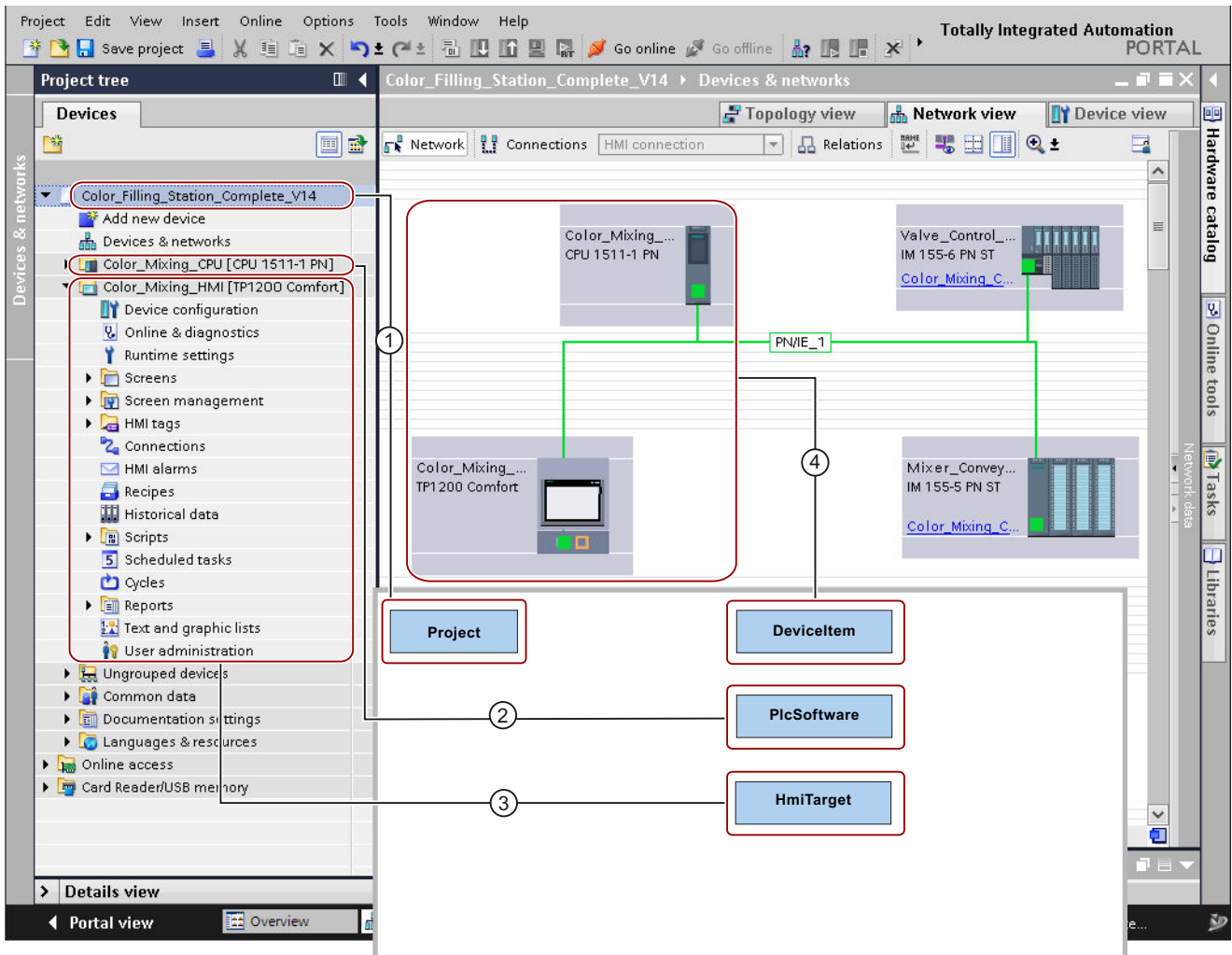
Nom	Type	Espace nom	Commentaire
PlcTag.Export(FileInfo path, ExportOptions options)	Méthode	Siemens.Engineering.SW.Tags	À présent, la valeur de l'attribut LogicalAddress est toujours exportée en abréviations internationales. Les abréviations allemandes sont encore acceptées pour l'importation.
PlcTag.LogicalAddress	Attribut	Siemens.Engineering.SW.Tags	À présent, la valeur de l'attribut LogicalAddress est toujours retournée en abréviations internationales. Les abréviations allemandes sont encore acceptées pour l'écriture.



Le schéma suivant illustre les objets disponibles sous PlcSoftware.

Relation entre TIA Portal et le modèle d'objet TIA Portal Openness

La figure suivante présente la relation entre le modèle d'objet et un projet dans TIA Portal :



- ① L'objet "Project" correspond à un projet ouvert dans TIA Portal.
- ② L'objet "PlcSoftware" est de type "SoftwareBase"④ et correspond à un API. Le contenu de l'objet correspond à un API dans la navigation du projet, avec accès à des objets tels que des blocs ou des variables API.
- ③ L'objet "HmiTarget" est de type "SoftwareBase"④ et correspond à un pupitre opérateur. Le contenu de l'objet correspond à un appareil IHM dans la navigation du projet, avec accès à des vues ou des variables IHM.
- ④ L'objet "DeviceItem" correspond à un objet dans l'éditeur "Appareils & réseaux". Un objet du type "DeviceItem" peut être aussi bien un châssis qu'un module enfilé.

9.1.3 Modifications apportées à la fonction du pilote

Introduction

Les modifications suivantes ont été apportées au modèle d'objet API V14 SP1 et sont pertinentes uniquement pour les utilisateurs qui ont utilisé la fonction pilote de HW Config dans V14 :

Modifications apportées aux types de TIA Portal Openness API

Type de TIA Portal Openness API	Nouveau type de TIA Portal Openness API
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

Modifications des Enums

Type de TIA Portal Openness API	Type de données	Nouveau type de TIA Portal Openness API	Type de données
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		distant	
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

Modifications des valeurs d'attributs de Siemens.Engineering.HW.IoConnector

Attribut	Type de données	Nouveau nom	Type de données
ProfinetUpdateTimeMode	ProfinetUpdateTimeMode	PnUpdateTimeAutoCalculation	Bool
ProfinetUpdateTime		PnUpdateTime	
AdaptUpdateTime		PnUpdateTimeAdaption	
WatchdogFactor		PNWatchdogFactor	
		DeviceNumber	String

Modifications des valeurs d'attributs de Siemens.Engineering.HW.ioController

Attribut	Type de données	Nouveau nom	Type de données
		DeviceNumber	String

Modifications des valeurs d'attributs de Siemens.Engineering.HW.Node

Attribut	Type de données	Nouveau nom	Type de données
HighestAddress		distant, disponible sur le sous-réseau uniquement	
TransmissionSpeed		distant, disponible sur le sous-réseau uniquement	
IsoProtocolUsed		UseIsoProtocol	
IpProtocolUsed		UseIpProtocol	
RouterAddressUsed		UseRouter	
PnDeviceNameAutoGenerated		PnDeviceNameAutoGeneration	
DeviceNumber		distant, déplacer sur IO-Connector / IO-Controller	

Modifications des valeurs d'attributs de Siemens.Engineering.HW.Subnet

Attribut	Type de données	Nouveau nom	Type de données
HighestAddress	Byte	HighestAddress	Int
CableConfiguration		PbCableConfiguration	
RepeaterCount		PbRepeaterCount	
CopperCableLength		PbCopperCableLength	
OpticalComponentCount		PbOpticalComponentCount	
OpticalCableLength		PbOpticalCableLength	
OpticalRingEnabled		PbOpticalRing	
OlmP12		PbOlmP12	
OlmG12		PbOlmP12	
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		PbDirectDateExchange	
MinimizeTslot		PbMinimizeTslotForSlaveFailure	
OptimizeCableConfig		PbOptimizeCableConfiguration	
CyclicDistribution		PbCyclicDistribution	
TslotInit		PbTslotInit	

Attribut	Type de données	Nouveau nom	Type de données
Tslot		PbTslot	
MinTsdr		PbMinTsdr	
MaxTsdr		PbMaxTsdr	
Tid1		PbTid1	
Tid2		PbTid2	
Trdy		PbTrdy	
Tset		PbTset	
Tqui		PbTqui	
Ttr		PbTtr	
TtrMs		distant	
TtrTypical		PbTtrTypical	
TtrTypicalMs		distant	
Watchdog		PbWatchdog	
WatchdogMs		distant	
Gap	Byte	PbGapFactor	Int
RetryLimit	Byte	PbRetryLimit	Int
IsochronMode		IsochronousMode	
AdditionalDevice		PbAdditionalPassivDeviceForIsochronousMode	
TotalDevice		PbTotalPassivDeviceForIsochronousMode	
DpCycleTimeAutoCalc		DpCycleMinTimeAutoCalculation	
TiToAutoCalc		IsochronousTiToAutoCalculation	
Ti		IsochronousTi	
To		IsochronousTo	

Modifications des valeurs d'attributs de Siemens.Engineering.Project

Attribut	Type de données	Nouveau nom	Type de données
.HwExtensions		.HwUtilities	

Modifications des valeurs d'attributs de Siemens.Engineering.HW.Baudrate

Attribut	Type de données	Nouveau nom	Type de données
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	

Principales modifications

9.1 Modifications importantes dans V14 SP1

Attribut	Type de données	Nouveau nom	Type de données
BaudRate.BAUD_6000000		BaudRate.BAUD6000000	
BaudRate.BAUD_12000000		BaudRate.BAUD12000000	

Modifications des valeurs d'attributs de Siemens.Engineering.HW.CableLength

Attribut	Type de données	Nouveau nom	Type de données
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	

Modifications des valeurs d'attributs de Siemens.Engineering.HW.ChanneloType

Attribut	Type de données	Nouveau nom	Type de données
ChanneloType.Unknown		ChanneloType.Complex	

Modifications des valeurs d'attributs de Siemens.Engineering.HW.IpProtocolSelection

Attribut	Type de données	Nouveau nom	Type de données
IpProtocolSelection.Address-Tailoring		IpProtocolSelection.VialoController	

Modifications des valeurs d'attributs de Siemens.Engineering.HW.TransmissionRateAndDuplex

Attribut	Type de données	Nouveau nom	Type de données
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	

Attribut	Type de données	Nouveau nom	Type de données
TransmissionRateAndDuplex.X1000Mbps_FullDuplex		TransmissionRateAndDuplex.X1000MbpsFullDuplex	
TransmissionRateAndDuplex.FO1000Mbps_FullDuplex_LD		TransmissionRateAndDuplex.FO1000MbpsFullDuplexLD	
TransmissionRateAndDuplex.FO1000Mbps_FullDuplex		TransmissionRateAndDuplex.FO1000MbpsFullDuplex	
TransmissionRateAndDuplex.TP1000Mbps_FullDuplex		TransmissionRateAndDuplex.TP1000MbpsFullDuplex	
TransmissionRateAndDuplex.FO10000Mbps_FullDuplex		TransmissionRateAndDuplex.FO10000MbpsFullDuplex	
TransmissionRateAndDuplex.FO100Mbps_FullDuplex_LD		TransmissionRateAndDuplex.FO100MbpsFullDuplexLD	
TransmissionRateAndDuplex.POFPCF100Mbps_FullDuplex_LD		TransmissionRateAndDuplex.POFPCF100MbpsFullDuplexLD	

9.1.4 Modifications de l'importation et de l'exportation

9.1.4.1 Modifications de l'importation et de l'exportation

Introduction

L'exportation et l'importation au moyen de la TIA Portal Openness API ont été améliorées dans V14 SP1 afin de pouvoir traiter les commentaires pour les éléments de tableau. Cela a rendu nécessaire un nouveau schéma. Deux versions de schéma sont désormais utilisées pour l'importation et l'exportation d'interfaces de blocs :

- Pour l'importation : le choix de la version de schéma utilisée s'opère sur la base de la plage de noms : <Sections xmlns=http://www.siemens.com/automation/Openness/SW/Interface/v2>
- Pour l'exportation : le choix de la version de schéma utilisée s'opère sur la base de la version du projet. Les projets V14 SP1 correspondent à la version 2, les projets V14, à la version 1.

9.1.4.2 Modifications dans l'API

Générer une source

Les méthodes suivantes ont été supprimées de ProgramBlocks :

- GenerateSourceFromBlocks
- GenerateSourceFromTypes

9.1 Modifications importantes dans V14 SP1

Les méthodes suivantes ont été complétées :

- GenerateSource à PlcExternalSourceSystemGroup

Exemple

```
// 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");

PlcBlocksSystemGroup blocksGroup = ...;
blocksGroup.GenerateSourceFromBlocks(blocks, fileInfo);
PlcTypesSystemGroup 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);
```

9.1.4.3 Extension des schémas

Extension des schémas pour les commentaires et les valeurs de démarrage

Les commentaires et les valeurs de démarrage sont enregistrés dans le nouvel élément "Subelement", qui se rapporte à l'élément de tableau possédant l'attribut "Path".

Subelement contient la valeur de démarrage et le commentaire relatifs à l'élément de tableau référencé. L'attribut "Path" de StartValue est supprimé du nouveau schéma.

Définition du schéma de "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>
```

Extension du type de membre :

```

<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>

```

Exemples :

Enregistrement de commentaires et valeurs de démarrage dans des tableaux simples :

```

<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>

```

Enregistrement de commentaires et valeurs de démarrage dans des tableaux de type de données UDT :

9.1 Modifications importantes dans V14 SP1

```
<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>
```

Enregistrement de commentaires et valeurs de démarrage dans des tableaux de type de données 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>
```

9.1.4.4 Modifications apportées au schéma

Nœud Access dans SW.PlcBlocks.Access.xsd

L'attribut Type du nœud Access a été déplacé vers les nœuds subordonnés de Access pour

- AbsoluteOffset – obligatoire
- Address – facultatif

```
<StlStatement UId="22">
  <StlToken Text="L" />
  <Access Scope="Address">
    <Address Area="Local" Type="Word" BitOffset="80" />
  </Access>
</StlStatement>
```

L'attribut Type de Constant a été remplacé par le nouveau sous-nœud ConstantType.

```
<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>
```

La valeur de l'attribut Scope dans Access a été renommée en TypedConstant quand ConstantValue contient une valeur qualifiée de type (exemple : int#10).

Constant ne possède aucun attribut Type quand ConstantValue contient une valeur qualifiée de type (exemple : int#10).

Les variables locales ne possèdent aucun nœud Address quand Scope est une LocalVariable.

Lorsqu'un Access est imbriqué dans un autre Access, quel que soit le niveau, seul l'Access extérieur a besoin d'un UId.

Nœud Address dans SW.PlcBlocks.Access.xsd

L'attribut BitOffset du nœud Address est désormais facultatif.

Le tableau suivant représente les modifications apportées aux déclarations pour exporter l'accès absolu :

Zone à partir de V14 SP1	Type	Numéro de bloc	Décalage de bit	Exemple
DB	Block_DB	obligatoire	interdit	OPN %DB12
DB	non trié	existant	obligatoire	%DB100.DBX10.3
DB	non trié	inexistant	obligatoire	%DB100.DBX10.3
L	non trié	interdit	obligatoire	%LW10.0

Zone à partir de V14 SP1	Type	Numéro de bloc	Décalage de bit	Exemple
I Q M	non trié	interdit	obligatoire	%I0.0 %Q0.0 %M0.0
T C	non trié	interdit	obligatoire	%T0 %C1
Block_FC Block_FC	Block_FC Block_FB	obligatoire	interdit	Appel %FB4, %DB5 Input_1 := %FC10 Appel %FB4, %DB5 Input_2 := %FB11
Entrée de périphérie	non trié	interdit	obligatoire	
Sortie de périphérie	non trié	interdit	obligatoire	

Nœud Area dans SW.PlcBlocks.Access.xsd

Le nœud Area possède désormais une liste de valeurs d'énumération simplifiée :

- LocalC et LocalIN deviennent Local
- DBc, DBv, DBr ont été supprimés.

Nœud CallInfo dans SW.PlcBlocks.Access.xsd

L'attribut Name du nœud CallInfo est désormais facultatif.

L'attribut BlockType du nœud CallInfo est désormais obligatoire.

+2.2.5 Appels de blocs utilisateurs

Nœud Constant dans SW.PlcBlocks.Access.xsd

Le nœud Constant référence le nœud CostantType avec minOccurs=0.

Le nœud Constant ne référence plus le nœud IntegerAttribute.

Nœud ConstantValue dans SW.PlcBlocks.Access.xsd

Le nœud ConstantValue reçoit un attribut informatif.

Nœud Instruction dans SW.PlcBlocks.Access.xsd

Le nœud Instruction référence le nœud Acces avec minOccurs=0.

Les attributs de paramètre Section, Type et TemplateReference ont été supprimés sur Instruction.

Nœud Parameter dans SW.PlcBlocks.Access.xsd

L'attribut SectionName du nœud Parameter est désormais facultatif.

Valeurs de Scope dans SW.PlcBlocks.Access.xsd

La liste des valeurs d'énumération de Scope a été complétée par :

- TypedConstant
- AddressConstant
- LiteralConstant
- AlarmConstant
- Address
- Statusword
- Expression
- Call
- CallWithType

Nœud Statusword dans SW.PlcBlocks.Access.xsd

La liste des valeurs d'énumération de Statusword a été complétée par :

- STW

Nœud ConstantType dans SW.PlcBlocks.Access.xsd

Le nouveau nœud ConstantType a été introduit avec l'attribut Informative facultatif.

Nœud CallRef dans SW.PlcBlocks.LADFBFD.xsd

Le nœud CallRef a été renommé en Call et ne possède plus de sous-nœud BooleanAttribute.

Nœud InstructionRef dans SW.PlcBlocks.LADFBFD.xsd

Le nœud InstructionRef a été remplacé par le nœud Part.

Nœud Part dans SW.PlcBlocks.LADFBFD.xsd

Le nouveau nœud ConstantType a été introduit et il remplace le nœud InstructionRef.

- Attributs : nom et version
- Sous-nœud : le sous-nœud Instruction disponible comme nouvelle sélection, en plus de Equation existant
- Ne possède ni sous-nœud BooleanAttribute ni attribut Gate.

Nœud Wire dans SW.PlcBlocks.LADFBFD.xsd

L'attribut Name du nœud Wire a été supprimé.

Nœud TemplateReference dans SW.PlcBlocks.LADFBDD.xsd

Le nœud TemplateReference a été supprimé.

Nœud StatementList dans SW.PlcBlocks.STL.xsd

Liste des valeurs d'énumération de StatementList (STL_TE) :

- L_STW a été supprimé
- T_STW a été supprimé

9.1.4.5 Modifications du comportement

Accès absolu

Dans V14, l'importation de l'accès absolu était annulée pour la plupart des combinaisons. A partir de V14 SP1, l'importation de l'accès absolu fonctionne pour les zones suivantes :

- entrée
- sortie
- mémentos
- temporisation, si pris en charge par l'API
- compteur, si pris en charge par l'API
- DB
- DI

En cas d'utilisation simultanée d'un accès symbolique et d'un accès absolu non rejetés après vérification du schéma ou du type de nœud, l'importation n'est réussie que quand les informations d'accès aux boîtes ont été résolues avec succès. Lorsque l'accès symbolique et l'accès absolu donnent des informations incohérentes, l'importation est refusée.

```
<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>
```


Accès direct via DB

A partir de V14 SP1, l'accès indirect via DB ne peut être importé que si vous avez défini les valeurs Offset, 'Type' et 'Symbole'.

```
...  
<Access Scope="LocalVariable" UId="21">  
  <Symbol>  
    <Component Name="Output_3" />  
    <AbsoluteOffset BitOffset="16" Type="Word" />  
  </Symbol>  
</Access>  
...
```

Informations symboliques et absolues pour l'accès local

Lors de l'importation d'un "accès symbolique", toutes les "informations pour l'accès absolu" possibles indiquées sont vérifiées, tant qu'elles ne sont pas marquées comme "informatives". A partir de V14 SP1, l'importation est annulée en cas d'incohérence des informations absolues.

Restrictions relatives à l'interface de bloc

Différentes restrictions sont à l'étude dans V14 SP1. Ces restrictions sont déjà connues des utilisateurs de l'éditeur d'interface de bloc. Chaque fois que l'éditeur d'interface de bloc renomme un paramètre en le complétant ou en augmentant sa valeur de '_1', le processus d'importation OPNS est annulé.

Par exemple, les restrictions suivantes sont à l'étude :

- Doubles noms de paramètre
- Noms de section erronés, y compris la 'section Return' des blocs FB
- Limitation de caractères

Tri des sections lors de l'importation

Lorsque le bloc appelé n'est pas disponible au moment de l'importation, c'est la définition de l'interface, côté appel, qui est utilisée pour afficher le bloc utilisateur appelé. Dans V14 SP1, les sections sont triées dans l'ordre dans lequel elles seraient affichées dans l'interface de bloc du bloc appelé, si celui-ci avait existé avec les mêmes paramètres.

Ordre d'affichage des sections des paramètres importés :

- entrée
- sortie

L'exemple en LIST xml suivant

9.1 Modifications importantes dans V14 SP1

```

<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>

```

entraîne :

<pre> CALL "Block_2" Input_1 := "Tag_1" Input_2 := "Tag_2" Output_1 := "Tag_3" Output_2 := "Tag_4" </pre>
--

Noms d'appelant univoques pour blocs utilisateurs

Les noms doivent être univoques dans TIA Portal. Autrement dit, une variable, par exemple, ne doit pas posséder le même nom qu'un bloc. Pour l'importation sous XML de la TIA Portal Openness API, cela signifie que lorsque le XML contient un appel de bloc utilisateur pour lequel le bloc appelé n'est pas disponible au moment de l'importation, le nom du bloc appelé doit être univoque par rapport à tous les noms existants dans le projet. Quand le nom du bloc appelé n'est pas univoque, le processus d'importation est annulé.

Dans l'exemple suivant, le processus d'importation est annulé parce que le nom du bloc appelé "Tag_1" est déjà utilisé pour une table de variables.

```

...
<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>
...

```

Dans l'exemple suivant, le processus d'importation est annulé parce que deux paramètres possèdent le même nom "Input1".

```

<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>

```

Appels de blocs de bibliothèque

Le fichier XML importé peut contenir des appels de blocs utilisateur. Ces blocs utilisateur sont identifiés par leur nom.

Les blocs utilisateur peuvent également appeler des éléments de bibliothèque. Ces éléments de bibliothèque peuvent être générés en tant que "Appels de blocs de bibliothèque". Comme les blocs de bibliothèque utilisent la même plage de nom que les blocs utilisateur, il est possible

9.1 Modifications importantes dans V14 SP1

que l'importation d'un appel de bloc utilisateur effectuée par le nom appelle l'implémentation d'un bloc utilisateur.

Avant V14 SP1, le processus d'importation essayait d'attribuer les paramètres entre l'appel du bloc utilisateur et l'appel du bloc d'instruction. Le processus d'importation était annulé ponctuellement, tous les paramètres incohérents étaient supprimés occasionnellement.

A partir de V14 SP1, l'appel du bloc utilisateur continue de trouver le bloc de bibliothèque, mais sans que l'appel ne devienne invalide.

Incohérence du type de bloc

Lorsque le XML contient un appel de bloc utilisateur du 'Block_1' possédant plus de paramètres que le FC correspondant n'en possède dans le projet, à partir de V14 SP1, l'importation définit pour le bloc appelé une nouvelle interface qui correspond à l'appel du bloc utilisateur existant dans le XML. A la prochaine compilation du bloc de programme, le système essaie d'actualiser l'appel.

Nouvelle étendue des fonctions pour constantes

A partir V14 SP1, différentes fonctions nouvelles ont été définies pour les constantes. L'importation ne peut réussir que si les valeurs dans le fichier XML correspondent à l'étendue des fonctions de la constante. Il se peut que le processus d'importation soit annulé quand toutes les informations spécifiées pour une constante ne correspondent pas à la constante existante.

```
...
  <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>
...
```

Remarques concernant les versions d'instruction

A partir de V14 SP1, vous ne pouvez importer que les versions d'instruction utilisables sur l'API dans lequel vous souhaitez importer. Si aucune version d'instruction n'est mentionnée dans le fichier XML, c'est la version sélectionnée dans l'API qui est utilisée. En CONT et en LOG, il n'existe aucune attribution de versions pour certains éléments représentés comme instructions. Ces éléments ne peuvent être importés que sans version.

```
...  
<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>  
...
```

ENO désactivé

Sur les automates S7-1200 et S7-1500, la fonction de désactivation d'ENO sert à désactiver les calculs de l'état de liaison d'ENO sollicitant le temps d'exécution.

A partir de V14 SP1, le mémento DisabledENO ne peut être importé sur les API qui prennent en charge cette nouvelle fonction.

```
...  
<Part Name="Add" UId="24" DisabledENO="false">  
  <TemplateValue Name="Card" Type="Cardinality">2</TemplateValue>  
  <TemplateValue Name="SrcType" Type="Type">Int</TemplateValue>  
</Part>  
...
```

Validation du type pour l'accès à la pile des données locales

A partir de V14 SP1, le processus d'importation est annulé lorsque le type n'est pas utilisé ou ne peut être attribué.

Validation d'Index-Ident

L'accès via l'Index est utilisable quand la fonction "Accès symbolique à la mémoire" est définie, par exemple accès local, accès global, accès indirect.

En cas d'utilisation d'une constante littérale comme index, les types entier signé ou non sont convertis en DINT. A partir de V14 SP1, le processus d'importation est annulé quand un type se situe en dehors de la plage indiquée.

Pour tout accès via l'Index, le système vérifie au préalable si 'accès via l'Index' peut vraiment être utilisé comme type d'accès. A partir de V14 SP1, le processus d'importation est annulé lorsque l'accès via l'Index défini ne peut pas être utilisé.

Tri dans l'ordre des éléments

A partir de V14 SP1, les éléments dans CONT et LOG sont triés dans l'ordre de génération des codes où l'exportation automatique est possible. Dans quelques rares cas, vous ne pouvez plus réimporter le fichier XML exporté. Dans ces cas, vous devez soit modifier le fichier XML ou supprimer et programmer de nouveau les réseaux correspondants. Toutefois, l'ordre des conducteurs et références reste peu fiable.

Constantes d'alarme

Avec V14 SP1, la vérification des compilations a été élargie à la validité des constantes d'alarme. Il peut arriver qu'en raison d'un XML importé dans V14 avec des constantes d'alarme défectueuses, un projet soit compilable dans V14 SP1. Dans ce cas, ouvrez le réseau pertinent dans l'éditeur CONT/LOG et supprimer l'opérande d'alarme réelle. L'éditeur crée automatiquement une nouvelle constante d'alarme valide.

```
<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>
```

Restrictions concernant les instances de blocs utilisateur et Motion Control

Dans V14, il était possible d'importer des appels de blocs utilisateur FC dans une instance et même de compiler ponctuellement ces appels.

A partir de V14 SP1, l'importation d'instances n'est possible que là où les instances sont prise en charge. Vous ne pourrez éventuellement plus compiler des projets existants avec des instances d'appels de blocs utilisateur et instructions. Dans ce cas, vous devez supprimer l'appel et le programmer de nouveau. Toute tentative visant à effectuer une actualisation d'appel ou tout autre type de réparation automatique est vouée à l'échec.

EnEno visible

Dans V14, c'est le memento ENENO qui déterminait si les entrées/sorties EN et ENO de 'InstructionRef' étaient utilisables ou pas.

A partir de V14 SP1, OPNS utilise pendant l'importation les entrées/sorties EN ou ENO en fonction de l'élément et du câblage. En raison de cette identification automatique, il convient de préciser toute utilisation différente des entrées/sorties EN et ENO. Il est très probable que seuls les boîtes des temporisations CEI et de compteurs CEI posent quelques problèmes.

Affectation d'UId

Avec V14 SP1, l'affectation de UId aux pièces, accès et conducteurs a changé. Les UId des expressions, CallInfo et opérandes doivent être univoques dans une unité de compilation. Du point de vue de TIA Portal, les UId présents dans la clé XML n'ont aucune signification supplémentaire à côté de l'identification d'un élément.

Contrôle des chaînes de caractères

Des contrôles plus stricts des guillemets, caractères génériques et caractères de contrôle de l'attribut Name sont effectués lors de l'importation de

- IntegerAttribute
- StringAttribute
- DateAttribute
- AutomaticTyped
- Component
- Invisible
- Label
- NameCon
- Negated
- TemplateValue
- CallInfo
- Instruction
- Parameter
- Part
- Step

Des contrôles plus stricts des caractères génériques et caractères de contrôle sont effectués lors de l'importation de

- Titres de blocs et réseaux
- Texte LineComment
- Chaînes de caractères de constantes (types STRING, WSTRING, CHAR, WCHAR)

Des contrôles plus stricts des caractères génériques et caractères de contrôle (tabulation et nouvelle ligne autorisées) sont effectués lors de l'importation de

- Commentaires et réseaux
- Attributs de chaînes de caractères
- Nœuds, définissant des textes multilingues comme Alarmtext, Comments
- Textes Token

Non-respect de la casse dans les modèles d'opération et paramètres

A partir de V14 SP1, les modèles d'opération pour les instructions et les paramètres d'appel ou d'instruction ne respectant pas la casse sont importés et corrigés automatiquement.

Le code suivant est importé, la valeur erronée "Eq" est corrigée en "EQ" et le paramètre erroné "iN1" est corrigé en "IN1" :

```
<StlStatement UId="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>
```

Multi-instances utilisées dans les appels

A partir de V14 SP1, le processus d'importation est annulé lorsque la multi-instance utilisée dans un appel est inexistante.

Le code suivant présente un exemple de XML dans lequel la multi-instance est correctement définie dans la section d'interface :

9.1 Modifications importantes dans V14 SP1

```

<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="&quot;Block_2&quot;" />
        </Section>
      </Sections>
    </Interface>
    ....
  <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>

```

Modèles de cardinalités dans LIST

Dans LIST, les modèles de cardinalités pour chaque instruction ont une valeur par défaut fixe, qui est la seule valeur valide. A partir de V14 SP1, le processus d'importation est annulé quand une valeur différente est utilisée pour la cardinalité.

Importer l'accès direct

A partir de V14 SP1, l'accès indirect ne peut être importé que là où il peut être compilé.

```

<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>

```

Importer des mots d'état

A partir de V14 SP1, vous ne pouvez importer le mot d'état qu'avec les instructions où il est pris en charge.

- L - mot d'état pris en charge : STW
- T - mot d'état pris en charge : STW
- A - mot d'état pris en charge : BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- AN - mot d'état pris en charge : BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- O - mot d'état pris en charge : BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- ON - mot d'état pris en charge : BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- X - mot d'état pris en charge : BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- XN - mot d'état pris en charge : BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU

Remarque

La plupart des mots d'état sont uniquement utiles pour les automates S7-300 et S7-400.

Instructions vides

Le processus d'importation est annulé quand une instruction ne possède aucun nœud `<StlStatement/>`. Dans une instruction vide, ajoutez le nœud `<StlToken Text="Empty_Line" />`.

Le processus d'importation est annulé quand une instruction vide dispose de commentaires. Pour une instruction possédant uniquement des commentaires, utilisez `<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>

```

9.1.4.6 Modifications apportées aux attributs de bloc

Modifications apportées aux attributs généraux

AutoNumber possède une nouvelle valeur par défaut (false) pour les OB classiques.

HeaderVersion possède un nouveau type System.Version (à la place de String).

9.1 Modifications importantes dans V14 SP1

IsKnowHowProtected s'applique également aux types de données définis par l'utilisateur.

ILibraryTypeInstance.ConnectedVersion, ILibraryTypeInstance.Dependencies, ILibraryTypeInstance.Dependents ont été supprimés du tableau des attributs généraux, car ils ne sont ni exportés dans XML ni accessibles via API.

MemoryLayout possède une nouvelle valeur par défaut : Standard pour API classiques et Optimized pour API Plus.

Number s'applique également aux types de données définis par l'utilisateur, est représenté dans XML et également accessible via API.

Modifications apportées aux attributs spécifiques

IsOnlyStoredInLoadMemory et IsWriteProtectedInAS sont désormais en lecture seule sur IDBofUDT, pour autant que l'UDT appartienne à un élément de bibliothèque système.

OfSystemLibElement et OfSystemLibVersion ne font plus partie des attributs généraux, mais appartiennent aux attributs spécifiques.

OfSystemLibVersion possède un nouveau type System.Version (à la place de String).

ParameterPassing reste accessible en lecture et en écriture pour les FC et les FB uniquement si

- ProgrammingLanguage est LIST
- MemoryLayout est Standard
- l'interface est vide.

GraphVersion possède un nouveau type System.Version (à la place de String).

Un nouvel attribut appelé ExtensionBlockName est introduit pour les FB écrits dans GRAPH (à partir de la version V4).

Un nouvel attribut appelé InvalidValuesAcquisition est introduit pour les FB écrits dans GRAPH (à partir de la version V4).

Un nouvel attribut appelé IsWriteProtected est introduit pour les blocs code.

DownloadWithoutReinit est désormais en lecture seule et valable également pour IDBofFBs.

Supervisions est désormais en lecture seule pour les IDBofFBs.

Modifications dans les énumérations

Les valeurs d'énumération de ProgrammingLanguage ont été modifiées de la manière suivante :

- Une valeur d'énumération F_CALL a été introduite.
- Une nouvelle valeur d'énumération Motion_DB a été introduite pour l'objet technologique Motion.
- GRAPH_SEQUENCE, GRAPH_ACTIONS, GRAPH_ADDINFOS ont été supprimés des énumérations. Ils ont été remplacés par GRAPH.

Les valeurs d'énumération de BlockType ont été modifiées de la manière suivante :

- Les valeurs OB, FC, DB, SFC ont été supprimées, car les énumérations correspondantes ne sont utilisées qu'avec l'attribut InstanceOfType.

9.2 Modifications importantes dans V14

9.2.1 Principales modifications du modèle d'objet

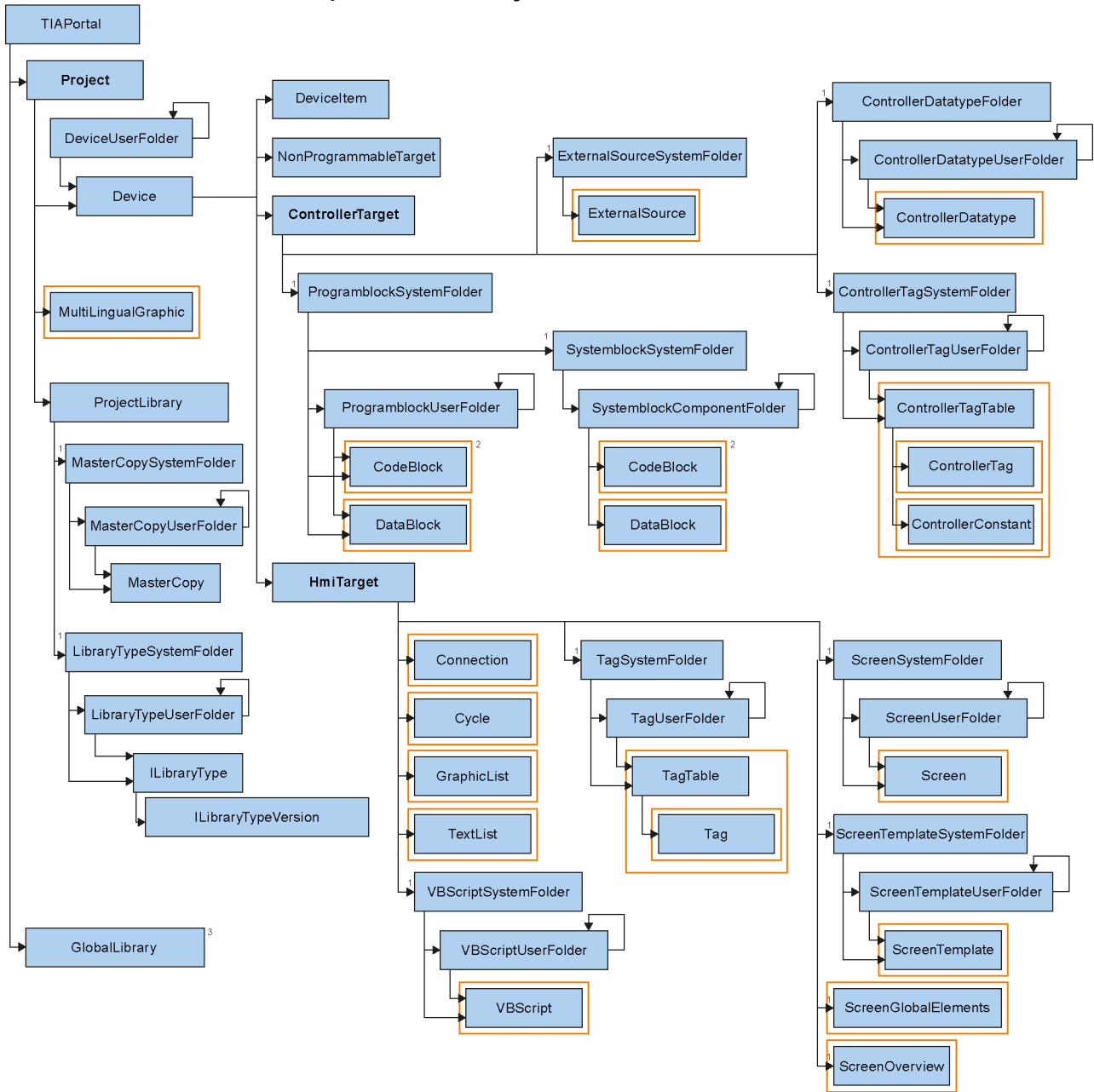
Modèle d'objet de TIA Portal Openness V13 SP1 et plus ancien

Pour permettre une comparaison entre l'ancien et le nouveau modèle d'objet de TIA Portal Openness, le diagramme ci-dessous décrit le modèle d'objet de TIA Portal V13 SP1.

Remarque

Le modèle objet décrit dans le diagramme est déconseillé. Vous trouverez des informations sur le modèle d'objet de TIA Portal Openness V14 SP1 sous Modèle d'objet TIA Portal Openness (Page 51)

Openness object model V13 SP1



9.2.2 Avant la mise à niveau d'une application vers TIA Portal Openness V14

Application

Les paramètres suivants doivent être modifiés avant la mise à niveau d'une application vers TIA Portal Openness V14 :

1. Les renvois vers l'API V14 doivent être adaptés en complétant les API TIA Portal Openness suivants :
 - `Siemens.Engineering`
 - `Siemens.Engineering.Hmi`
2. Mettre à niveau le .Net-Framework de Visual Studio vers la version 4.6.1.
3. Actualisez la méthode `AssemblyResolve` en modifiant le nouveau chemin d'installation de TIA Portal.
 - Si vous travaillez à partir du fichier d'enregistrement, modifiez la nouvelle clé comme dans l'exemple suivant :

```
"HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\_InstalledSW
\TIAP14\TIA_Opns\..."
```
 - Si vous travaillez avec le fichier de configuration d'application, adaptez les chemins au nouveau chemin d'installation.

9.2.3 Principales modifications de chaîne de caractères

Introduction

Les modifications suivantes ont été apportées à TIA Portal Openness V14 et il se peut que cela ait des effets sur vos applications existantes :

Modification	Adaptation nécessaire du code de programme
Les méthodes de compilation ont été modifiées.	<p>Modifiez les méthodes de compilation comme dans l'exemple suivant :</p> <ul style="list-style-type: none"> • TIA Portal Openness V13 SP1(obsolète) : <code>controllerTarget.Compile(CompilerOptions. Software, BuildOptions.Rebuild);</code> • TIA Portal Openness V14: <code>plcSoftware.GetService<ICompilable>().Compile();</code>
De nouveaux espaces de noms ont été ajoutés.	<ol style="list-style-type: none"> 1. Ajoutez les instructions d'espaces de noms suivantes : <code>Siemens.Engineering.SW.Blocks;</code> <code>Siemens.Engineering.SW.ExternalSources;</code> <code>Siemens.Engineering.SW.Tags;</code> <code>Siemens.Engineering.SW.Types;</code> 2. Supprimez l'instruction d'espace de nom <code>using ControllerTarget = Siemens.Engineering.HW.ControllerTarget.</code> 3. Compilez l'application.
<code>ControllerTarget</code> a été remplacé par <code>PlcSoftware</code> et la fonctionnalité a été modifiée dans certains cas.	<ol style="list-style-type: none"> 1. Vérifiez les exemples de code dans la documentation faisant partie de votre fonctionnalité d'application. 2. Actualisez le code du programme de votre application TIA Portal Openness d'après l'exemple suivant : <ul style="list-style-type: none"> – TIA Portal Openness V13 SP1(obsolète) : <code>ControllerTarget controllerTarget = deviceItem as ControllerTarget</code> – TIA Portal Openness V14: <code>PlcSoftware plcSoftware = deviceItem.GetService<SoftwareContainer>().Software as PlcSoftware</code> 3. Compilez l'application.

Modification	Adaptation nécessaire du code de programme
Des objets ont été remplacés.	<p>1. Recherchez et remplacez les objets suivants :</p> <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> <p>2. Compilez l'application.</p>
Les agrégations ont été remplacées par des compositions.	<p>1. Remplacez chaque Aggregation dans votre code par Composition comme dans les exemples suivants :</p> <pre> ProjectAggregation = ProjectComposition IDeviceAggregation = IDeviceComposition TagTableAggregation = TagTableComposition CycleAggregation = CycleComposition GraphicListAggregation = GraphicListComposition TextListAggregation = TextListComposition ConnectionAggregation = ConnectionComposition MultiLingualGraphicAggregation = MultiLingualGraphicComposition UpdateCheckResultMessageAggregation = UpdateCheckResultMessageComposition </pre> <p>2. Compilez l'application.</p>
Les dossiers ont été remplacés par des groupes dans toutes les relations, sauf pour concerne les pupitres opérateurs.	<p>1. Excepté les sections de code concernant les pupitres opérateurs, remplacez chaque Folder dans votre code de programme par Group.</p> <p>2. Compilez l'application.</p>

Modification	Adaptation nécessaire du code de programme
La méthode <code>GetAttributeNames</code> a été remplacée par la méthode <code>GetAttributeInfos</code> .	<ol style="list-style-type: none"> 1. Pour déterminer les attributs, utilisez <pre>IList<EngineeringAttributeInfo> IEngineeringObject.GetAttributeInfos (AttributeAccessMode attributeAccessMode); .</pre> 2. Compilez l'application. Pour plus d'informations, référez-vous à Déterminer la structure et les attributs de l'objet (Page 109).
La méthode <code>Close</code> de fermeture d'un projet a été modifiée.	<ol style="list-style-type: none"> 1. Remplacez <pre>project.Close (CloseMode.PromptIfModified) ; par project.Close ();</pre> 2. Compilez l'application. Pour plus d'informations, référez-vous à Fermer un projet (Page 120).
L'accès simultané a été remplacé par l'accès exclusif et des transactions exclusives.	<ol style="list-style-type: none"> 1. Remplacez l'accès simultané par l'accès exclusif et des transactions exclusives comme dans les exemples suivants : <ul style="list-style-type: none"> - TIA Portal Openness V13 SP1 (obsolète) : <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> 2. Compilez l'application. Pour plus d'informations, voir Exclusive access (Page 90) et Traitement des transactions (Page 92).

Modification	Adaptation nécessaire du code de programme
L'accès en ligne à la CPU a été modifié	<ol style="list-style-type: none"> Modifiez l'accès en ligne à la CPU comme dans les exemples suivants : <ul style="list-style-type: none"> TIA Portal Openness V13 SP1 (obsolète) : <pre>((IOOnline)controllerTarget).GoOffline();</pre> TIA Portal Openness V14: <pre>((DeviceItem)plcSoftware.Parent.Parent).GetService<OnlineProvider>().GoOffline();</pre> Compilez l'application.
La configuration matérielle a été modifiée	<ol style="list-style-type: none"> Modifiez la configuration matérielle : <pre>Device.Elements = Device.Items</pre> Retirez les attributs de matériel suivants : <ul style="list-style-type: none"> Device.InternalDeviceItem Device.SubType Compilez l'application.

Voir aussi

Traitement des exceptions (Page 367)

Nouveautés de TIA Portal Openness (Page 21)

Etablissement d'une connexion au portail TIA (Page 74)

9.2.4 Importation de fichiers créés avec TIA Portal Openness V13 SP1 et des versions antérieures

Application

Si vous essayez d'importer des fichiers qui ont été générés avec TIA Portal Openness V13 SP1 ou une version antérieure, une exception d'incompatibilité est déclenchée. Motif : des modifications dans les variables IHM et les écrans IHM. Les tableaux suivants montrent les principales modifications d'attributs. Vous trouverez des informations détaillées dans le chapitre "Créer les vues > Utilisation des objets et de groupes d'objets > Utilisation des objets > Configuration de plages" dans l'aide en ligne de TIA Portal :

Modifications de variables IHM

Le tableau suivant montre les principales modifications d'attributs de variables IHM :

Attributs supprimés	Attributs ajoutés
RangeMaximumType	LimitUpper2Type.
RangeMaximum	LimitUpper2.
RangeMinimumType	LimitLower2Type.
RangeMinimum	LimitLower2.
	LimitUpper1Type
	LimitUpper1
	LimitLower1Type
	LimitLower1

Modifications d'éléments graphiques IHM

Le tableau suivant montre les principales modifications d'attributs de curseur :

Attributs supprimés	Attributs ajoutés
	RangeLower1Color
	RangeLower1Enabled
	RangeLower2Color
	RangeLower2Enabled
	RangeNormalColor
	RangeNormalEnabled
	RangeUpper1Color
	RangeUpper1Enabled
	RangeUpper2Color
	RangeUpper2Enabled
	ScalePosition
	ShowLimitLines
	ShowLimitMarkers
	ShowLimitRanges

9.2 Modifications importantes dans V14

Le tableau suivant montre les principales modifications d'attributs de plage de mesure :

Attributs supprimés	Attributs ajoutés
DangerRangeColor	RangeLower1Color
DangerRangeStart	RangeLower1Enabled
DangerRangeVisible	RangeLower2Color
WarningRangeColor	RangeLower2Enabled
WarningRangeStart	RangeNormalColor
WarningRangeVisible	RangeNormalEnabled
	RangeUpper1Color
	RangeUpper1Enabled
	RangeUpper1Start
	RangeUpper2Color
	RangeUpper2Enabled
	RangeUpper2Start

Le tableau suivant montre les principales modifications d'attributs de bargraphe :

Attributs supprimés	Attributs ajoutés
AlarmLowerLimitColor	RangeLower1Color
AlarmUpperLimitColor	RangeLower1Enabled
	RangeLower2Color
	RangeLower2Enabled
	RangeNormalColor
	RangeNormalEnabled
	RangeUpper1Color
	RangeUpper1Enabled
	RangeUpper2Color
	RangeUpper2Enabled

Index

A

- Accéder
 - Copie maître dans une bibliothèque de projet, 145
- Acquittement d'événements système piloté par le programme, 82
- API
 - Comparaison avec état réel, 274
 - Comparer, 274
 - Déterminer statut, 254
 - Etablir une liaison en ligne, 276
 - Interrompre la liaison en ligne/déconnecter, 276

B

- Bibliothèque
 - Accéder à des dossiers, 132
 - Déterminer les versions de types d'instances, 151
 - Fonctions, 125
- Bibliothèque de projet
 - Accéder, 126, 128
 - Accéder aux copies maîtres, 145
- Bibliothèque globale
 - Accéder, 126, 128
- Bloc
 - Créer un groupe, 283
 - Exporter, 448
 - Générer une source, 290
 - Importer, 494
 - Interroger des informations, 281
 - Supprimer, 282
 - Supprimer un groupe, 286
- Bloc de programme
 - Supprimer, 282

C

- Compilation
 - Groupe d'objets technologiques, 301
 - Logiciel, 113
 - Matériel, 113
 - Objet technologique, 300
- Condition d'édition
 - Votre application Openness et le TIA Portal se trouvent sur le même ordinateur, 46

- Configuration
 - Votre application Openness et le TIA Portal se trouvent sur différents ordinateurs, 45
- Connecter
 - Axe synchrone avec des valeurs pilote, 330
 - Came, 327
 - Codeur, 325
 - Codeur à un bloc de données, 313
 - Codeurs pour les entraînements analogiques à l'adresse matérielle, 311
 - Codeurs pour les PROFIdrives à l'adresse matérielle, 309
 - Entraînements, 320
 - Entraînements analogiques à l'adresse matérielle, 310
 - Entraînements analogiques avec un bloc de données, 312
 - Palpeur de mesure, 328
 - Piste de came, 327
 - PROFIdrives à l'adresse matérielle, 308
 - PROFIdrives avec un bloc de données, 312
- Connexion au portail TIA
 - Fermer, 84
- Copie maître
 - Copie, 151
 - Copier le contenu dans dossier de projet, 148
- Copier
 - Contenu d'une copie maître dans dossier de projet, 148
 - Copie maître, 151
- Copies maître
 - Supprimer, 157
- Coupure de la connexion au portail TIA, 84
- Création
 - Créer des dossiers de vues personnalisés, 244
 - Dossier personnalisé pour les tables des variables API, 338
 - Dossiers personnalisés pour variables IHM, 249
 - Groupe pour bloc, 283
 - Sous-dossiers personnalisés pour scripts, 252
- Créer
 - Came, 314
 - Objet technologique, 299
 - Palpeur de mesure, 314
 - Piste de came, 314

D

- Démarrer
 - Editeur de bloc, 294
 - Editeur de variables, 335
- Dossier
 - Supprimer, 157

E

- Ecrire
 - Paramètres d'un objet technologique, 306
- Éditeur "Appareils et réseaux"
 - Ouvrir, 160
- Editeur de bloc
 - Démarrer, 294
- Editeur de variables
 - Démarrer, 335
- Enregistrer le projet, 119
- Énumérer
 - Blocs, 279
 - Dossiers Blocs personnalisés, 278
 - Dossiers personnalisés pour variables API, 337
 - Objet technologique, 301
 - Paramètres d'un objet technologique, 303
 - Sous-dossier système, 287
 - Tables de variables API, 339
 - Toutes les variables d'une table de variables, 250
 - Variables API, 343
- Énumérer
 - Appareils, 211, 214
 - Éléments d'appareil, 227
 - Textes multilingues, 107, 112
- Énumérer des appareils, 211, 214
- Énumérer des éléments d'appareil, 227
- Énumérer des textes multilingues, 107, 112
- Établir une liaison à TIA Portal, 74
- Exceptions
 - En cas d'accès au portail TIA avec des API publics, 367
- Exemple d'application Public API, 67
- Exemple de programme, 49
- Exportation/importation
 - Utilisation, 34
- Exporter
 - Bloc, 448
 - Type de données utilisateur, 448
 - Une seule variable ou constante d'une table de variables API, 497

F

- Fichier d'exportation
 - Contenu, 376
 - Structure de base, 387, 505, 509
 - Structure du fichier XML, 387, 509
- Fichier XML
 - Editer, 375
 - Exportation, 376
- Fonctions, 49
 - API, 278, 279, 281, 287, 338, 339, 342, 344, 345, 496, 497, 498
 - Constantes API, 345
 - Créer des dossiers de vues personnalisés, 244
 - Créer les sous-dossiers personnalisés pour scripts, 252
 - Créer un dossier personnalisé pour tables des variables API, 338
 - Définir un dossier système, 287
 - Enregistrer le projet, 119
 - Énumérer des appareils, 211, 214
 - Énumérer des blocs, 279
 - Énumérer des éléments d'appareil, 227
 - Énumérer des tables de variables API dans des dossiers, 339
 - Énumérer des textes multilingues, 107, 112
 - Énumérer des variables API, 343
 - Énumérer le dossier Blocs personnalisé, 278
 - Énumérer le sous-dossier système, 287
 - Énumérer les dossiers personnalisés pour variables API, 337
 - Énumérer les variables d'une table de variables IHM, 250
 - Exemple d'application Public API, 67
 - Exporter une variable ou une constante d'une table de variables API, 497
 - Fermer un projet, 120
 - Généralités, 74, 82, 84
 - Générer des dossiers personnalisés pour variables IHM, 249
 - IHM, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253
 - Importer une table de variables API, 496
 - Importer une variable dans une table des variables API, 498
 - Interroger la famille du bloc, 281
 - Interroger la version du bloc, 281
 - Interroger l'attribut "Consistency" d'un bloc, 281
 - Interroger l'auteur du bloc, 281
 - Interroger le dossier "Blocs de programme", 278
 - Interroger le nom de bloc, 281

Interroger le numéro de bloc, 281
 Interroger le titre du bloc, 281
 Interroger le type de bloc, 281
 Interroger les informations d'une table de variables API, 340
 Interroger l'horodatage d'un bloc, 281
 Interroger PLC Target et HMI Target, 161
 Interroger un dossier système pour variables API, 336
 Lecture de l'heure des dernières modifications dans une table des variables API, 342
 Limitation aux projets de TIA Portal V13, 99
 Ouvrir un projet, 99
 Paramètres généraux de TIA Portal, 104
 Projets, 99, 104, 107, 112, 118, 119, 120, 161, 211, 214, 227, 336, 337, 339, 340, 343
 Suppression de cycle, 247
 Suppression de la bibliothèque de graphiques, 118
 Suppression de la liaison, 249
 Suppression de la liste de textes, 247
 Suppression de la table des variables, 251
 Suppression de la table des variables API, 342
 Suppression de toutes les vues, 246
 Suppression d'un modèle de vue, 245
 Suppression d'une liste de graphiques, 248
 Suppression d'une variable d'une table des variables API, 344
 Suppression d'une vue, 244
 Supprimer les scripts VB d'un dossier, 253
 Supprimer un dossier personnalisé pour tables de variables API, 339
 Supprimer une variable d'une table des variables, 251

G

Générer
 Source à partir d'un bloc, 290
 Source à partir d'un type de données utilisateur, 290
 Groupe d'objets technologiques
 Compilation, 301

H

Hiérarchie des objets matériels du modèle d'objet, 64

I

Importation/exportation
 Editer un fichier XML, 375
 Importer
 Bloc, 494
 Tables de variables API, 496
 Type de données utilisateur, 500
 Une seule variable dans une table des variables API, 498
 Importer/Exporter
 AML-GUID stables, 522
 API, 448, 454, 491
 Appareils et modules Round-Trip, 522
 Définir le comportement d'importation à l'aide de codes de programme, 377
 Domaine d'application, 373
 Données du projet, 380, 381
 Exportation de commentaires multilingues, 544, 560
 Exportation de cycles, 390
 Exportation de données de configuration, 375
 Exportation de vues contextuelles, 425
 Exportation d'une vue Slide-in, 427
 Exporter aussi les valeurs standard, 376
 Exporter des blocs avec protection know-how, 454
 Exporter des blocs sans protection know-how, 448
 Exporter des blocs système, 491
 Exporter des commentaires multilingues, 407
 Exporter des connexions, 407
 Exporter des listes de textes, 402
 Exporter des modèles de vue, 421
 Exporter des scripts VB, 399, 400
 Exporter des tables de variables API, 495
 Exporter des tables de variables IHM, 392
 Exporter des variables,
 Exporter la variable sélectionnée, 396
 Exporter les listes de graphiques, 406
 Exporter les valeurs modifiées uniquement, 376
 Exporter les vues d'un appareil IHM, 413
 Exporter l'image avec une instance de bloc d'affichage, 430
 Exporter tous les graphiques d'un projet, 380
 exporter tous les modèles de vue, 420
 Exporter une fenêtre permanente, 418
 Exporter une variable d'une table de variables, 396
 Exporter une vue à partir d'un dossier de vues, 414

- Format d'exportation, 373
- Formats XML avancés pour l'exportation/
importation de listes de textes, 404
- Graphiques, 379
- IHM, 390, 391, 392, 395, 396, 397, 398, 399, 400,
401, 402, 403, 404, 406, 408, 409, 413, 414, 416,
418, 419, 420, 421, 423, 425, 426, 427, 429, 430,
432, 495
- Importation de commentaires multilingues, 544,
560
- Importation de connexions, 408
- Importation de données de configuration, 377
- Importation de vues contextuelles, 426
- Importation d'une vue Slide-in, 429
- Importer des commentaires multilingues,
- Importer des cycles, 391
- Importer des graphiques dans un projet, 381
- Importer des listes de textes, 403
- Importer des modèles de vue, 423
- Importer des scripts VB, 401
- Importer des variables,
- Importer des vues dans un appareil IHM, 416
- Importer les listes de graphiques, 406
- Importer l'image avec une instance de bloc
d'affichage, 432
- Importer une fenêtre permanente, 419
- Importer une table de variables dans un dossier
de variables, 395
- Importer une variable HMI dans une table de
variables, 397
- Limiter les exportations aux valeurs
modifiées, 376
- Marche à suivre pour l'importation, 378
- notions de base, 371
- Objets d'AML, 505
- Objets exportables, 371
- Objets graphiques exportables, 409
- Objets importables, 371
- Paramétrage de l'exportation, 375
- Particularités des variables IHM intégrées, 398
- Restrictions, 373
- Structure des données, 387, 509
- Volume d'exportation, 375
- Installation
 - Ajouter un utilisateur au groupe d'utilisateurs, 26
 - Étapes standard pour l'accès au portail TIA, 32
 - TIA Openness V13 complément, 25
 - Vérifier l'authentification d'accès, 26
- Installation du complément, 25
- Instances
 - Définir une version de type, 151

- Interrogation
 - Objet technologique, 298
- Interroger
 - Informations du bloc, 281
 - Informations du type de données utilisateur, 281

L

- Liaison à TIA Portal
 - Configuration, 74
- Lire
 - Heure des dernières modifications dans une table
des variables API, 342
 - Paramètres d'un objet technologique, 305
- Logiciel
 - Compilation, 113

M

- Matériel
 - Compilation, 113
- Modèle d'objet, 51

O

- Objet technologique, 295
 - Compilation, 300
 - Créer, 299
 - Enumérer, 301
 - Interrogation, 298
 - Rechercher, 302
 - Supprimer, 300
 - Types de données, 297
- Objets
 - Objets exportables, 371
 - Objets importables, 371
- Objets graphiques exportables, 409
- Ouverture d'un projet, 99
- Ouvrir
 - Éditeur "Appareils et réseaux", 160

P

- Paramètres
 - Comptage, 334
 - Easy Motion Control, 334
 - Régulation PID, 333
 - S7-1500 Motion Control, 316
- Paramètres d'un objet technologique
 - Ecrire, 306

- Enumérer, 303
- Lire, 305
- Rechercher, 304
- Paramètres généraux de TIA Portal, 104
- Particularités des variables IHM du type de données "UDT", 399
- Projet
 - Enregistrer, 119
 - Fermer, 120
 - Interroger HMI Targets, 161
 - Interroger PLC Targets, 161
 - Interroger un type d'appareil, 161
 - Ouvrir, 99

R

- Rechercher
 - Objet technologique, 302
 - Paramètres d'un objet technologique, 304
- Requêtes
 - Attribut "Consistency" d'un bloc, 281
 - Auteur du bloc, 281
 - Dossier "Blocs de programme", 278
 - Dossier système pour variables API, 336
 - Famille de bloc, 281
 - Horodatage d'un bloc, 281
 - Informations d'une table de variables API, 340
 - Nom de bloc, 281
 - Numéro de bloc, 281
 - Titre du bloc, 281
 - Trouver, 287
 - Type de bloc, 281
 - Version du bloc, 281

S

- Siemens.Engineering, 40
- Siemens.Engineering.Hmi, 40
- Siemens.Engineering.Hmi.Communication, 40
- Siemens.Engineering.Hmi.Cycle, 40
- Siemens.Engineering.Hmi.Globalization, 40
- Siemens.Engineering.Hmi.RuntimeScripting, 40
- Siemens.Engineering.Hmi.Screen, 40
- Siemens.Engineering.Hmi.Tag, 40
- Siemens.Engineering.Hmi.TextGraphicList, 40
- Siemens.Engineering.HW, 40
- Siemens.Engineering.SW, 40
- Statut (API)
 - Déterminer, 254
- Structure de base d'un fichier d'exportation, 387, 509

- Structure de base d'un fichier d'exportation
 - AML, 505
- Structure des données d'exportation, 387, 505, 509
- Suppression
 - Bibliothèque de graphiques, 118
 - Certaines variables d'une table des variables, 251
 - Connexion, 249
 - Cycle, 247
 - Liste de graphiques, 248
 - Liste de textes, 247
 - Modèle de vue, 245
 - Script VB d'un dossier, 253
 - Table des variables, 251
 - Toutes les vues, 246
 - Une seule variable d'une table des variables
 - API, 344
 - Vue, 244
- Supprimer
 - Bloc, 282
 - Bloc de programme, 282
 - Constantes API, 345
 - Dossier personnalisé pour les tables des variables API, 339
 - Groupe pour bloc, 286
 - Objet technologique, 300
 - Supprimer une table de variables API d'un dossier, 342
 - Type de données utilisateur, 293

T

- TIA Portal Openness, 43
 - Accès, 33
 - Ajouter un utilisateur au groupe d'utilisateurs, 26
 - API publique, 49
 - Concepts sous-jacents à la vérification de l'identité d'objet, 168
 - Concepts sous-jacents d'affectations, 167
 - Concepts sous-jacents pour la manipulation d'exceptions, 367
 - Conditions, 23
 - Configuration, 45
 - Droits d'accès, 26
 - Etapes standard pour l'accès au portail TIA, 32
 - Etendue des fonctions, 43
 - Exportation/importation, 34
 - Fonctions, 49
 - Introduction, 43
 - Notions de base sur la composition, 167
 - Savoir-faire nécessaire de l'utilisateur, 23

- Tâches typiques, 33
- Vue d'ensemble de la programmation, 49
- Trouver
 - Came, 314
 - Palpeur de mesure, 314
 - Piste de came, 314
- Type de données utilisateur
 - Exporter, 448
 - Générer une source, 290
 - Importer, 500
 - Interroger des informations, 281
 - Supprimer, 293
- Types
 - Supprimer, 157
- Types de données
 - Objet technologique, 297

V

- Variables IHM du type de données "UDT", 399
- Variables IHM intégrées, 398
- Vue d'ensemble de la programmation, 49