Tipps und Tricks zur Projektierung von Faceplates mit WinCC Unified

SIMATIC WinCC Unified V18 / Unified Comfort Panels

https://support.industry.siemens.com/cs/ww/de/view/109812366

Siemens Industry Online Support

Rechtliche Hinweise

Nutzung der Anwendungsbeispiele

In den Anwendungsbeispielen wird die Lösung von Automatisierungsaufgaben im Zusammenspiel mehrerer Komponenten in Form von Text, Grafiken und/oder Software-Bausteinen beispielhaft dargestellt. Die Anwendungsbeispiele sind ein kostenloser Service der Siemens AG und/oder einer Tochtergesellschaft der Siemens AG („Siemens“). Sie sind unverbindlich und erheben keinen Anspruch auf Vollständigkeit und Funktionsfähigkeit hinsichtlich Konfiguration und Ausstattung. Die Anwendungsbeispiele stellen keine kundenspezifischen Lösungen dar, sondern bieten lediglich Hilfestellung bei typischen Aufgabenstellungen. Sie sind selbst für den sachgemäßen und sicheren Betrieb der Produkte innerhalb der geltenden Vorschriften verantwortlich und müssen dazu die Funktion des jeweiligen Anwendungsbeispiels überprüfen und auf Ihre Anlage individuell anpassen.
Sie erhalten von Siemens das nicht ausschließliche, nicht unterlizenzierbare und nicht übertragbare Recht, die Anwendungsbeispiele durch fachlich geschultes Personal zu nutzen. Jede Änderung an den Anwendungsbeispielen erfolgt auf Ihre Verantwortung. Die Weitergabe an Dritte oder Vervielfältigung der Anwendungsbeispiele oder von Auszügen daraus ist nur in Kombination mit Ihren eigenen Produkten gestattet. Die Anwendungsbeispiele unterliegen nicht zwingend den üblichen Tests und Qualitätsprüfungen eines kostenpflichtigen Produkts, können Funktions- und Leistungsmängel enthalten und mit Fehlern behaftet sein. Sie sind verpflichtet, die Nutzung so zu gestalten, dass eventuelle Fehlfunktionen nicht zu Sachschäden oder der Verletzung von Personen führen.

Haftungsausschluss

Siemens schließt seine Haftung, gleich aus welchem Rechtsgrund, insbesondere für die Verwendbarkeit, Verfügbarkeit, Vollständigkeit und Mangelfreiheit der Anwendungsbeispiele, sowie dazugehöriger Hinweise, Projektierungs- und Leistungsdaten und dadurch verursachte Schäden aus. Dies gilt nicht, soweit Siemens zwingend haftet, z.B. nach dem Produkthaftungsgesetz, in Fällen des Vorsatzes, der groben Fahrlässigkeit, wegen der schuldhaften Verletzung des Lebens, des Körpers oder der Gesundheit, bei Nichteinhaltung einer übernommenen Garantie, wegen des arglistigen Verschweigens eines Mangels oder wegen der schuldhaften Verletzung wesentlicher Vertragspflichten. Der Schadensersatzanspruch für die Verletzung wesentlicher Vertragspflichten ist jedoch auf den vertragstypischen, vorhersehbaren Schaden begrenzt, soweit nicht Vorsatz oder grobe Fahrlässigkeit vorliegen oder wegen der Verletzung des Lebens, des Körpers oder der Gesundheit gehaftet wird. Eine Änderung der Beweislast zu Ihrem Nachteil ist mit den vorstehenden Regelungen nicht verbunden. Von in diesem Zusammenhang bestehenden oder entstehenden Ansprüchen Dritter stellen Sie Siemens frei, soweit Siemens nicht gesetzlich zwingend haftet.
Durch Nutzung der Anwendungsbeispiele erkennen Sie an, dass Siemens über die beschriebene Haftungsregelung hinaus nicht für etwaige Schäden haftbar gemacht werden kann.

Weitere Hinweise

Siemens behält sich das Recht vor, Änderungen an den Anwendungsbeispielen jederzeit ohne Ankündigung durchzuführen. Bei Abweichungen zwischen den Vorschlägen in den Anwendungsbeispielen und anderen Siemens Publikationen, wie z. B. Katalogen, hat der Inhalt der anderen Dokumentation Vorrang.
Ergänzend gelten die Siemens Nutzungsbedingungen (https://support.industry.siemens.com).

Securityhinweise

Siemens bietet Produkte und Lösungen mit Industrial Security-Funktionen an, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen.
Um Anlagen, Systeme, Maschinen und Netzwerke gegen Cyber-Bedrohungen zu sichern, ist es erforderlich, ein ganzheitliches Industrial Security-Konzept zu implementieren (und kontinuierlich aufrechtzuerhalten), das dem aktuellen Stand der Technik entspricht. Die Produkte und Lösungen von Siemens formen einen Bestandteil eines solchen Konzepts.
Die Kunden sind dafür verantwortlich, unbefugten Zugriff auf ihre Anlagen, Systeme, Maschinen und Netzwerke zu verhindern. Diese Systeme, Maschinen und Komponenten sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn und soweit dies notwendig ist und nur wenn entsprechende Schutzmaßnahmen (z.B. Firewalls und/oder Netzwerksegmentierung) ergriffen wurden.
Weiterführende Informationen zu möglichen Schutzmaßnahmen im Bereich Industrial Security finden Sie unter https://www.siemens.com/industrialsecurity.
Die Produkte und Lösungen von Siemens werden ständig weiterentwickelt, um sie noch sicherer zu machen. Siemens empfiehlt ausdrücklich, Produkt-Updates anzuwenden, sobald sie zur Verfügung stehen und immer nur die aktuellen Produktversionen zu verwenden. Die Verwendung veralteter oder nicht mehr unterstützter Versionen kann das Risiko von Cyber-Bedrohungen erhöhen.
Um stets über Produkt-Updates informiert zu sein, abonnieren Sie den Siemens Industrial Security RSS Feed unter https://www.siemens.com/cert.

Inhaltsverzeichnis

1 Einführung

1.1 Überblick

In der modernen Industrie spielt das Thema der Standardisierung eine immer größere Rolle. Speziell im Anlagen- und Maschinenbau ist die Anforderung an ein einheitliches Bedien- und Steuerungskonzept stark angestiegen. Das liegt z. B. an kurzen Einarbeitungszeiten für Bediener der Anlage oder an einer hohen Bediensicherheit.

Auch bei der Wartung von Anlagen ist ein einheitliches Bedien- und Steuerungskonzept wichtig. Bei Störungen, Wartungsarbeiten bzw. bei der Erweiterung einer Anlage können dadurch die Stillstandszeiten verringert werden. Im Serienmaschinenbau kommen projektierte Funktionen wie z. B. Antrieb Ein/Aus, Umschaltung Automatikbetrieb/Handbetrieb usw. immer wieder vor. Diese Funktionen müssen in der Regel nur an die entsprechende Maschine und deren Steuerungsvariablen angepasst werden. Hier bietet es sich beim Engineering an, auf vorgefertigte Objekte zurückzugreifen, um dadurch Zeit und Kosten zu sparen.

Im HMI-Bereich können Sie für diesen Zweck sogenannte „Faceplates“ (Bildbausteine) verwenden.

In diesem Dokument erfahren Sie, wie Sie verschiedene UseCases und deren Projektierungsmöglichkeiten von Faceplates im WinCC Unified Umfeld projektieren und verwenden.

1.2 Voraussetzungen

Für eine gute Nachvollziehbarkeit des Anwendungsbeispiels empfehlen wir Ihnen die beiden folgenden Online Support Beiträge oder SITRAIN-Kurse.

Damit sind Sie im allgemeinen Umgang mit WinCC Unified und Faceplates vertraut.

- Dokument SIMATIC HMI WinCC Unified Getting Started
- Anwendungsbeispiel SIMATIC WinCC Unified - Tipps und Tricks zur Skripterstellung (JavaScript)
- SITRAIN-Systemkurs: WinCC Unified & Unified Comfort Panels (Beitrags-ID: 109773211)
- SITRAIN-Aufbaukurs: SIMATIC WinCC Unified für PC-Systeme (Beitrags-ID: 109781323)

1.3 Verwendete Komponenten

Dieses Anwendungsbeispiel wurde mit folgenden Softwarekomponenten erstellt:

Tabelle 1-1

Komponente	Anzahl	Artikelnummer	Hinweis
SIMATIC WinCC Unified V18 PC Engineering (TIA Portal)	1	6AV2153-2FB01-8LA5	Engineering System (jede Lizenzgröße)
SIMATIC WinCC Unified V18 PC Runtime	1	6AV2154-2FB01-8BA0	PC Runtime (jede Lizenzgröße)

Dieses Anwendungsbeispiel besteht aus folgenden Komponenten:

Tabelle 1-2

Komponente	Dateiname	Hinweis
Dieses Dokument	109812366_Unified_FaceplatesTT_de_V10.docx	
TIA Portal Projekt	109812366_Unified_FaceplatesTT_V18.zip	Erstellt mit V18

2 Faceplate in Faceplate

2.1 Allgemeines

Ab TIA Portal V18 haben Sie die Möglichkeit Faceplates ineinander zu verschachteln und die zugehörigen Schnittstellen hierarchisch weiterzugeben ("Vererbung"). Sie können Anwenderdatentypen (engl. User Data Types – UDTs) einer Steuerung und deren interne Verschachtelungen identisch in den Datenstrukturen der Faceplates abbilden und so Ihre Projektierung bei geringerem Aufwand komplexer gestalten – Stichwort Standardisierung.

Hinweis

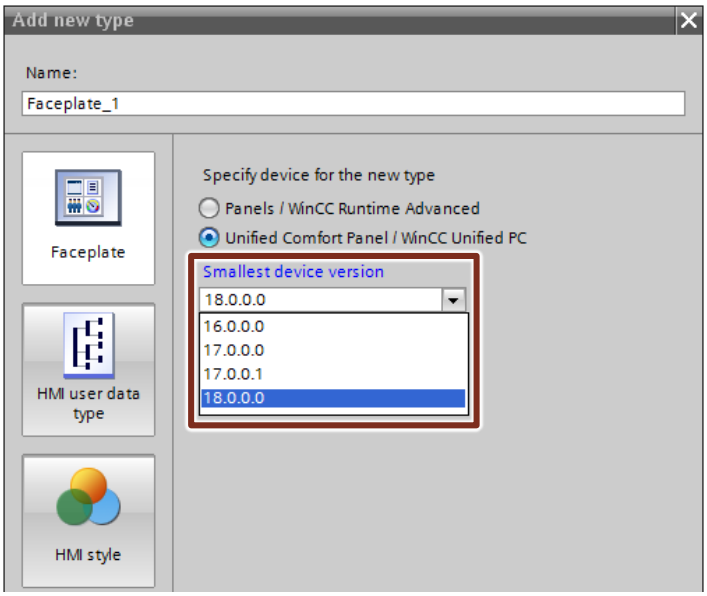
Technisch ist die Tiefe der Verschachtelung von Faceplates nicht begrenzt.
Wir empfehlen eine Verschachtelungstiefe von max. acht (8) Faceplates nicht zu überschreiten.

2.2 Niedrigste Geräteversion

TIA Portal V18 ermöglicht Ihnen das Festlegen der niedrigsten Runtime-Version, die auf einem HMI-Gerät installiert sein muss, um eine korrekte Darstellung der verwendeten Faceplates (WinCC Unified) zu gewährleisten.
Ab TIA Portal V16 wurden mit jeder Version neue Funktionen für WinCC Unified Faceplates eingeführt. Da diese Funktionen zur Laufzeit nicht abwärtskompatibel sind, ist es nötig, bei Verwendung bestimmter Funktionen die passende Runtime-Version auf dem Endgerät auszuführen.

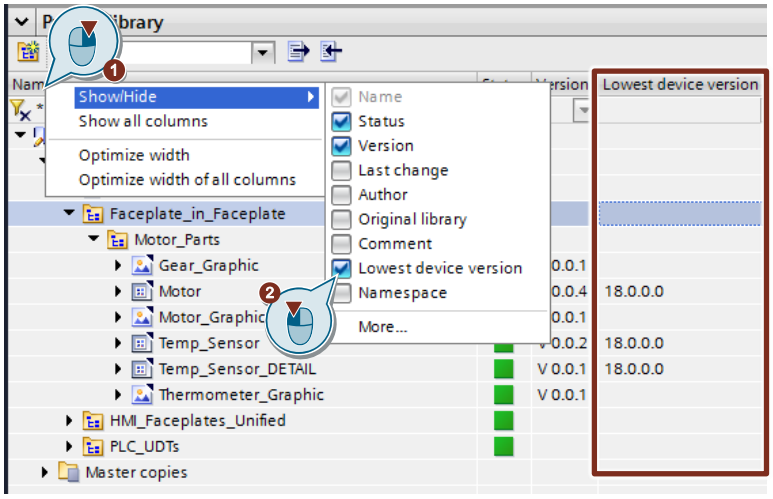
Um dies sicherzustellen, können Sie bei der Erstellung eines neuen Faceplate-Typs über das Drop-Down-Menü "niedrigste Geräteversion" die niedrigste Runtime-Version festlegen, die für eine fehlerfreie Ausführung benötigt wird.
In der Regel handelt es sich dabei um die Version, mit der die Funktion eingeführt wurde.

Abbildung 2-1



Um die konfigurierte niedrigste Geräteversion in der Bibliothek anzuzeigen, klicken Sie mit der rechten Maustaste auf den Spaltenkopf (1) und wählen Sie unter dem Eintrag "Anzeigen/Verbergen" die entsprechende Option aus (2).

Abbildung 2-2



Hinweis

Weitere Informationen und eine Zuordnung von Funktionen zu Versionen finden Sie im TIA Portal V18 Handbuch im Kapitel "Niedrigste Geräteversion eines Faceplate-Typs" (https://support.industry.siemens.com/cs/de/de/view/109813308/160875372683)

2.3 Beispiel – Verschachtelung von Faceplates

In diesem Beispiel sollen zwei Faceplates ineinander verschachtelt werden, sowie ein weiteres Faceplate als PopUp aus einem Faceplate heraus geöffnet werden. Für die Veranschaulichung dient ein Motor (äußeres Faceplate) mit zugehörigem Temperatursensor (inneres Faceplate), sowie einer Detailanzeige der spezifischen Temperaturen (PopUp Faceplate).

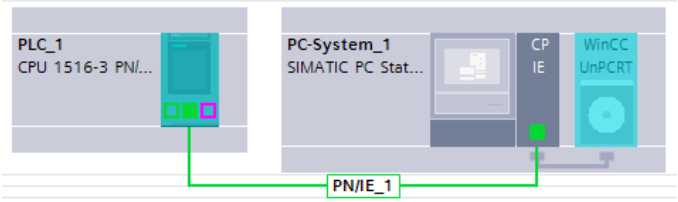
Voraussetzungen

Das Anlegen und Einrichten der nötigen Geräte und Verbindungen in TIA Portal wird in diesem Beispiel nicht behandelt.
Folgende Voraussetzungen sind notwendig, um den Einstieg in das Beispiel nachvollziehen zu können:

- Ein Projekt wurde angelegt und in der Projekt-Ansicht geöffnet
- Eine Steuerung wurde angelegt (hier: CPU 1516-3 PN/DP (6ES7 516-3AN00-0AB0, V1.8)
- Ein Unified HMI-Gerät wurde angelegt (hier: SIMATIC WinCC Unified PC Station (6AV2 155-xxxxx-xxxx, Version 18.0.0.0, mit "IE Allgemein" Kommunikationsmodul))

- Es wurde eine Netzwerkverbindung (PN/IE) zwischen beiden Geräten angelegt
- Es wurde eine HMI-Verbindung zwischen beiden Geräten angelegt

Abbildung 2-3



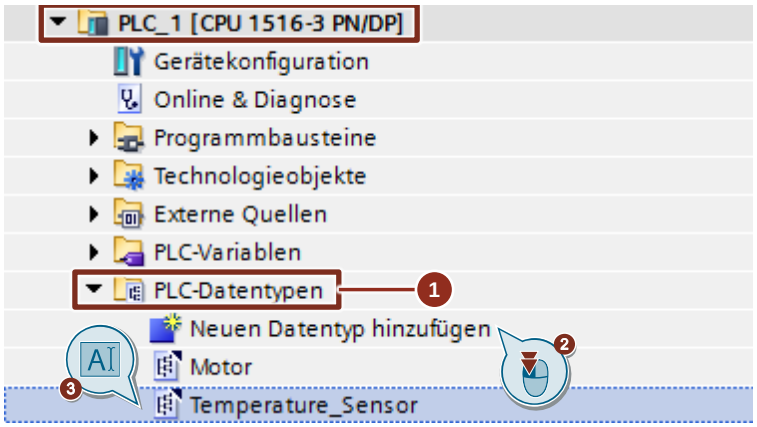
2.3.1 Anlegen von Anwenderdatentypen in der Steuerung

Temperatursensor

Da auch die Anwenderdatentypen verschachtelt werden sollen, legen Sie als Erstes den Anwenderdatentyp für das innere Faceplate (Temperatursensor) an.

1. Navigieren Sie in der Steuerung (PLC) zum Ordner "PLC-Datentypen"
2. Legen Sie durch einen Doppelklick auf "Neuen Datentyp hinzufügen" einen neuen Anwenderdatentyp an. Dieser wird automatisch im Editor geöffnet. Geben Sie dem Anwenderdatentyp einen sprechenden Namen ("Temperature_Sensor").

Abbildung 2-4

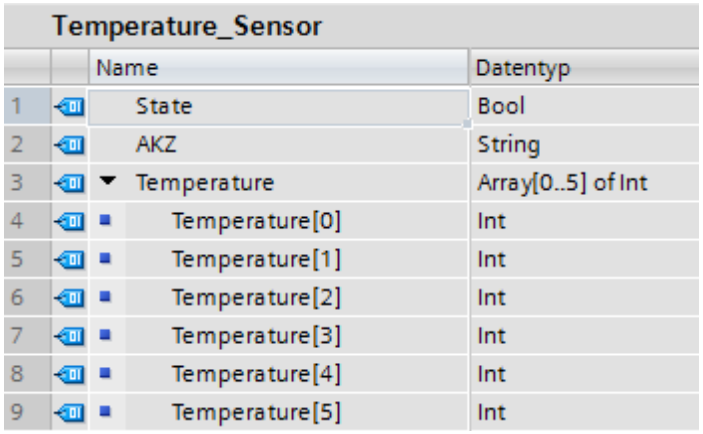


3. Legen Sie im Editor des Datentyps über "Hinzufügen" folgende Datenstruktur an:

Tabelle 2-1

Name	Datentyp
State	Bool
AKZ	String
Temperature	Array[0..5] of Int

Abbildung 2-5



Temperature_Sensor

	Name	Datentyp
1	State	Bool
2	AKZ	String
3	Temperature	Array[0..5] of Int
4	Temperature[0]	Int
5	Temperature[1]	Int
6	Temperature[2]	Int
7	Temperature[3]	Int
8	Temperature[4]	Int
9	Temperature[5]	Int

Motor

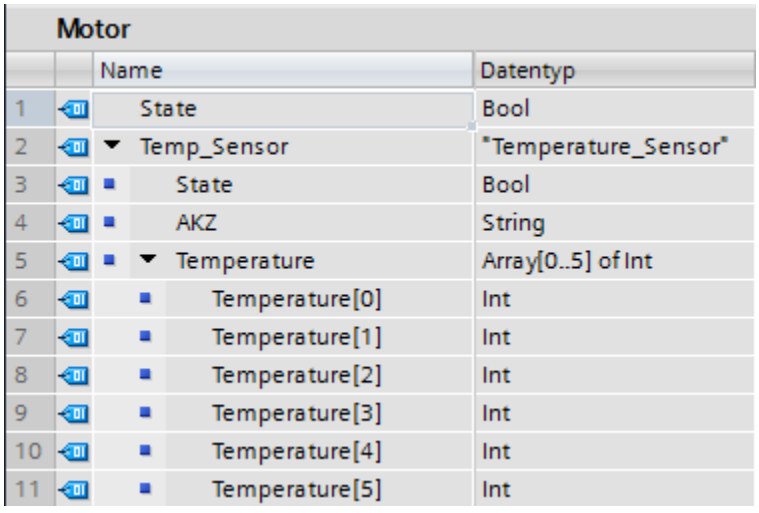
Wiederholen Sie die Schritte 1.-3. des vorherigen Abschnitts (Temperatursensor), benennen Sie den Datentyp "Motor" und legen Sie folgende Datenstruktur an:

Tabelle 2-2

Name	Datentyp
State	Bool
Temp_Sensor	"Temperature_Sensor"

Mit dem Datentyp "Temperature_Sensor" betten Sie die Datenstruktur des zuvor angelegten Anwenderdatentyps (Temperature_Sensor) als eigenes Element ein und erzeugen so eine Verschachtelung der beiden Anwenderdatentypen.

Abbildung 2-6



Motor

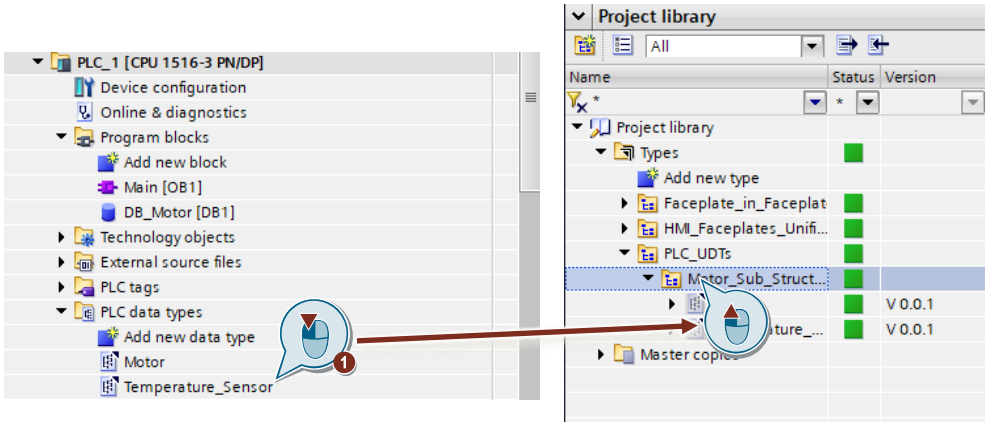
	Name	Datentyp
1	State	Bool
2	Temp_Sensor	"Temperature_Sensor"
3	State	Bool
4	AKZ	String
5	Temperature	Array[0..5] of Int
6	Temperature[0]	Int
7	Temperature[1]	Int
8	Temperature[2]	Int
9	Temperature[3]	Int
10	Temperature[4]	Int
11	Temperature[5]	Int

Anlegen von Typen in der Projektbibliothek

Um die erstellten Anwenderdatentypen auch in Verbindung mit HMI-Objekten (wie z. B. Faceplates) verwenden zu können, müssen Sie diese in der Projektbibliothek versionieren.

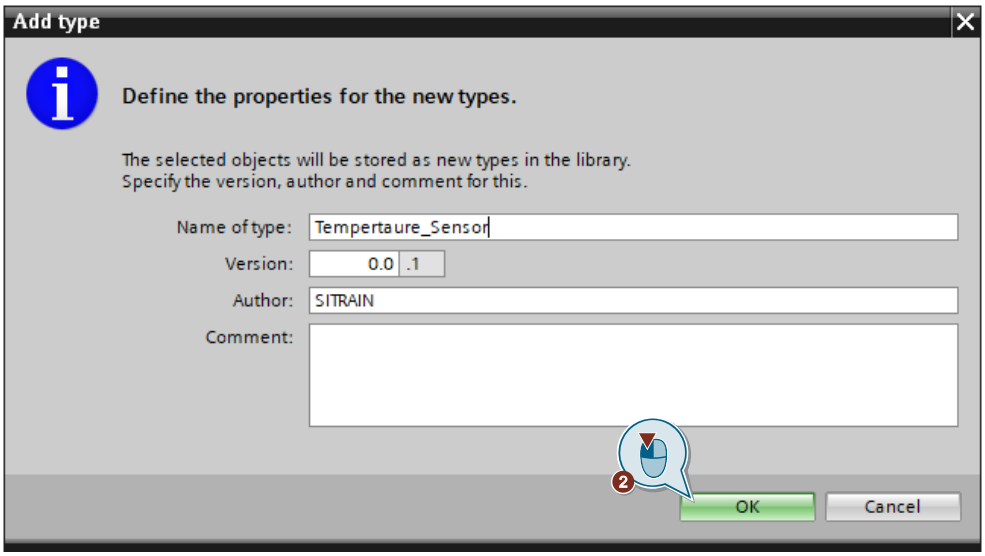
1. Selektieren Sie den Anwenderdatentyp "Temperature_Sensor" und ziehen Sie ihn per Drag&Drop in den Ordner "Typen" der Projektbibliothek (hier: Unterordner "PLC_UDTs > Motor_Sub_Structures")

Abbildung 2-7



2. Der Dialog "Typ hinzufügen" öffnet sich. Bestätigen Sie diesen mit einem Klick auf "OK"

Abbildung 2-8



3. Wiederholen Sie die Schritte 1. und 2. für den Anwenderdatentyp "Motor"

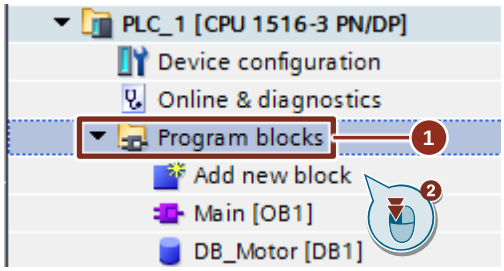
Ergebnis

Sie haben innerhalb der Steuerung eine verschachtelte Datenstruktur erstellt und diese in der Projektbibliothek versioniert. Im nächsten Schritt übertragen Sie diese Struktur auf einen Datenbaustein, der für die Kommunikation zum HMI-Gerät benötigt wird.

2.3.2 Übertragen der Struktur an einen Datenbaustein

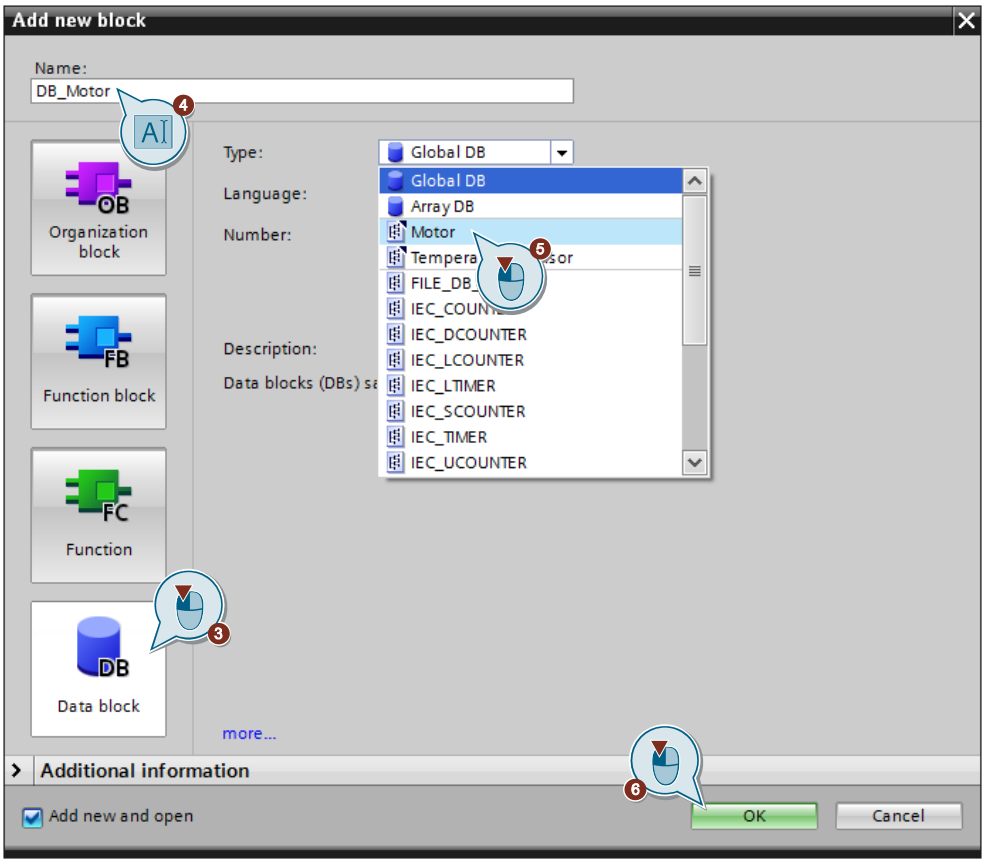
1. Navigieren Sie in der Steuerung zum Ordner "Programmbausteine".
2. Öffnen Sie durch einen Doppelklick auf "Neuen Baustein hinzufügen" den gleichnamigen Dialog für die Erstellung eines Programmbausteins.

Abbildung 2-9



3. Wählen Sie im Dialogfenster den Typ "Datenbaustein".
4. Ändern Sie den Namen des Datenbausteins ("DB_Motor").
5. Wählen Sie im Drop-Down-Menü den Typ „Motor“.
6. Erzeugen Sie den Datenbaustein mit einem Klick auf "OK".

Abbildung 2-10



Ergebnis

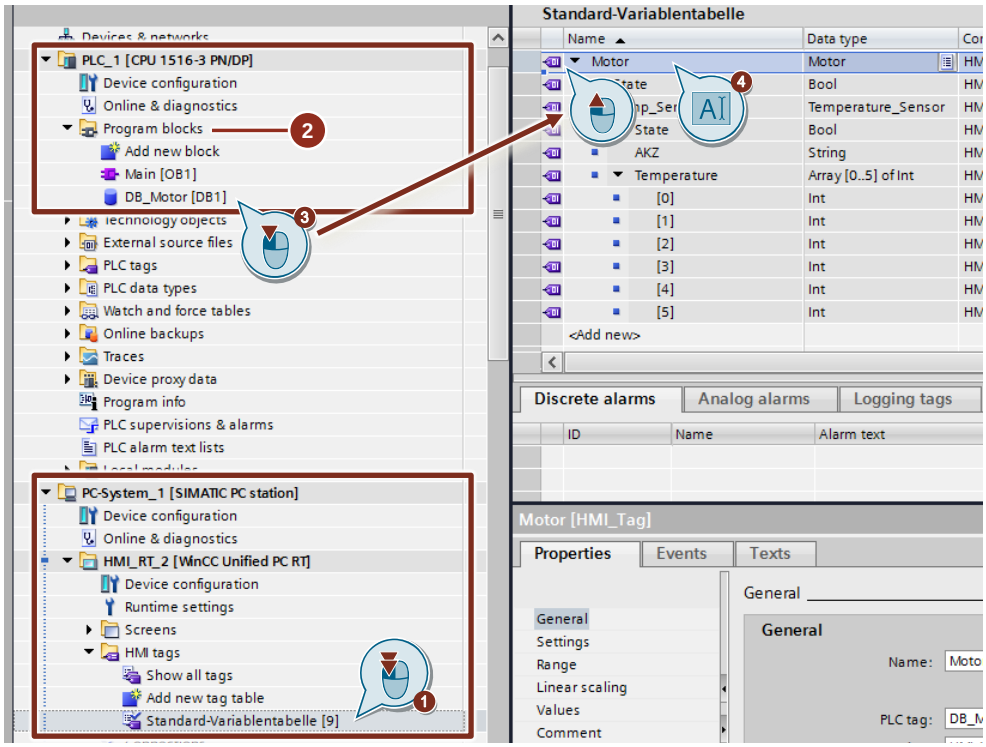
Sie haben den PLC-Datenbaustein "DB_Motor" mit der Struktur des Anwenderdatentyps "Motor" angelegt. Im nächsten Schritt legen Sie HMI-Variablen derselben Struktur an und erzeugen eine Verbindung zum Datenbaustein, welche für die Kommunikation zwischen den Geräten benötigt wird.

2.3.3 Anlegen der HMI-Variablen

In diesem Schritt werden die HMI-Variablen erzeugt, die an die Schnittstellen der Faceplates angebunden werden und die Kommunikation zwischen PLC und HMI ermöglichen.

1. Navigieren Sie im HMI-Gerät zum Ordner "HMI-Variablen" und öffnen Sie mit einem Doppelklick die "Standard-Variablentabelle".
2. Navigieren Sie in der PLC zum Ordner "Programmbausteine".
3. Ziehen Sie per Drag & Drop den Datenbaustein "DB_Motor" in die geöffnete Standard-Variablentabelle.
4. Die Struktur des Datenbausteins wird identisch an die HMI-Variablen weitergegeben. Ändern Sie den Namen der Variablen-Struktur von "DB_Motor" zu "Motor".

Abbildung 2-11



Ergebnis

Sie haben die Datenstruktur des PLC-Datenbausteins in HMI-Variablen übertragen und eine Verbindung zwischen den PLC- und HMI-Variablen hergestellt. Im nächsten Schritt erstellen und konfigurieren Sie die Faceplates.

2.3.4 Erstellen der Faceplates

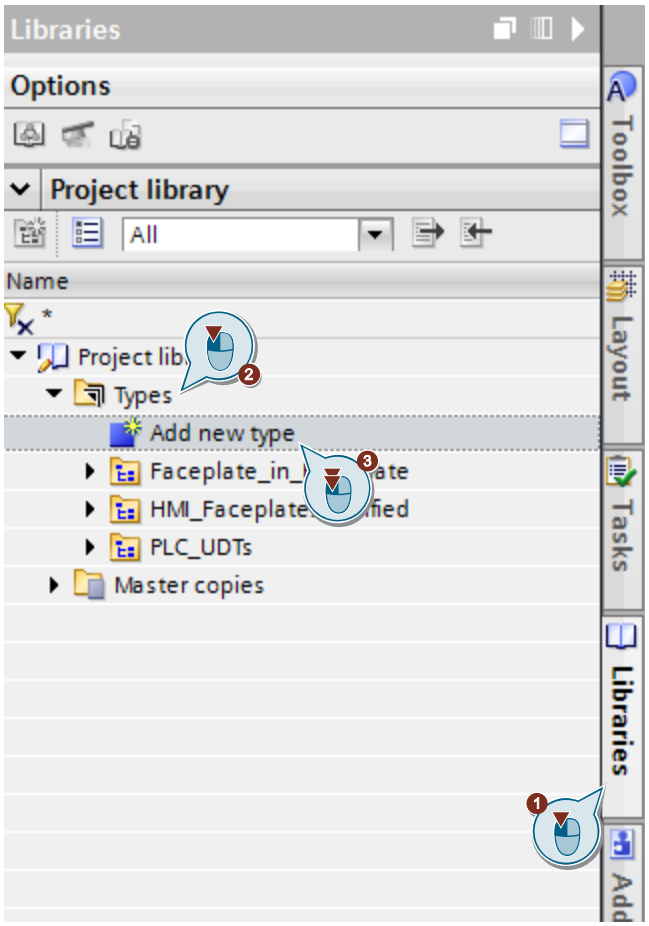
In diesem Abschnitt erstellen Sie die Faceplates, konfigurieren sie und verschachteln diese ineinander. Dabei arbeiten Sie von "innen" (PopUp-Faceplate, Temperatur Details) über das eingebettete Faceplate (Temperatursensor) nach "außen" (Motor)

2.3.4.1 PopUp-Faceplate "Temp_Sensor_DETAIL"

Anlegen des Faceplate-Typs

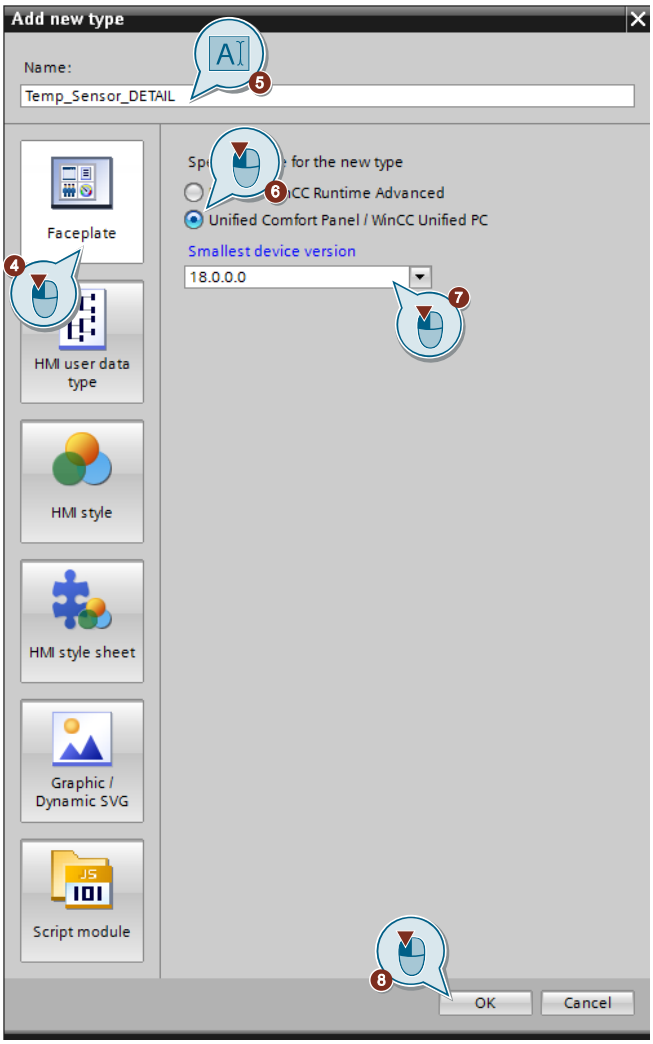
1. Wechseln Sie in den Reiter "Bibliotheken“.
2. Navigieren Sie zum Ordner "Projektbibliothek -> Typen".
3. Öffnen Sie mit einem Doppelklick auf "Neuen Typ hinzufügen" den gleichnamigen Editor.

Abbildung 2-12



4. Wählen Sie im Dialog "Neuen Typ hinzufügen" den Typ "HMI Bildbaustein" aus.
5. Ändern Sie den Namen zu "Temp_Sensor_DETAIL".
6. Wählen Sie als Gerät "Unified Comfort Panels / WinCC Unified PC" aus.
7. Wählen Sie für die niedrigste Geräteversion "18.0.0.0" aus.
8. Bestätigen Sie die Auswahl mit einem Klick auf "OK". Das Faceplate wird erzeugt und im Editor geöffnet.

Abbildung 2-13

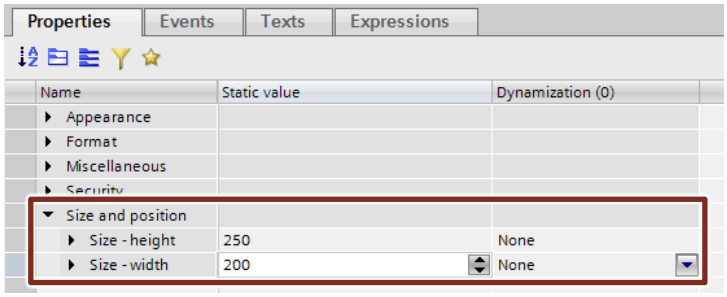


Anpassen der Visualisierung

Ändern Sie in den Eigenschaften des Faceplates die Größe auf folgende Werte:

- Größe – Breite: 200
- Größe – Höhe: 250

Abbildung 2-14



Fügen Sie nun die in Tabelle 2-3 aufgelisteten Basisobjekte hinzu und projektieren Sie damit die folgende Visualisierung:

Abbildung 2-15

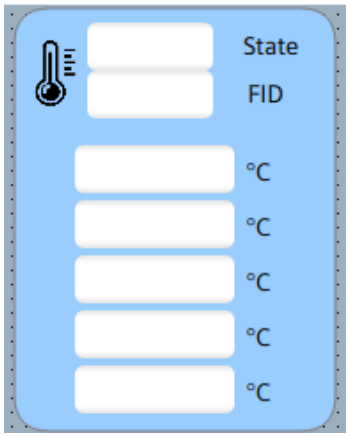


Tabelle 2-3

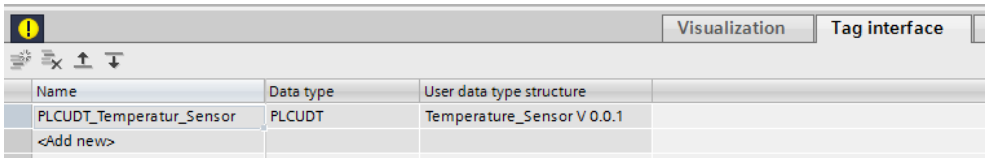
Basisobjekt	Anzahl	Anmerkungen
Rechteck	1	Hintergrundfarbe: 153; 204; 255 Eckenradius: 20
EA-Feld	7	Modus: Ausgabe
Textfeld	7	-
Grafikanzeige	1	Grafik: "Thermometer_Graphic V 0.0.1" (In Bibliothek des Anwendungsbeispiels enthalten)

Konfigurieren der Variablen Schnittstelle

Wechseln Sie in den Reiter "Variablen Schnittstelle", legen Sie mit einem Klick auf "<Hinzufügen>" eine neue Schnittstelle an und vergeben Sie folgende Parameter:

- Name: "PLCUDT_Temperatur_Sensor"
- Datentyp: "PLCUDT"
- Anwenderdatentyp-Struktur: "Temperature_Sensor V 0.0.1"

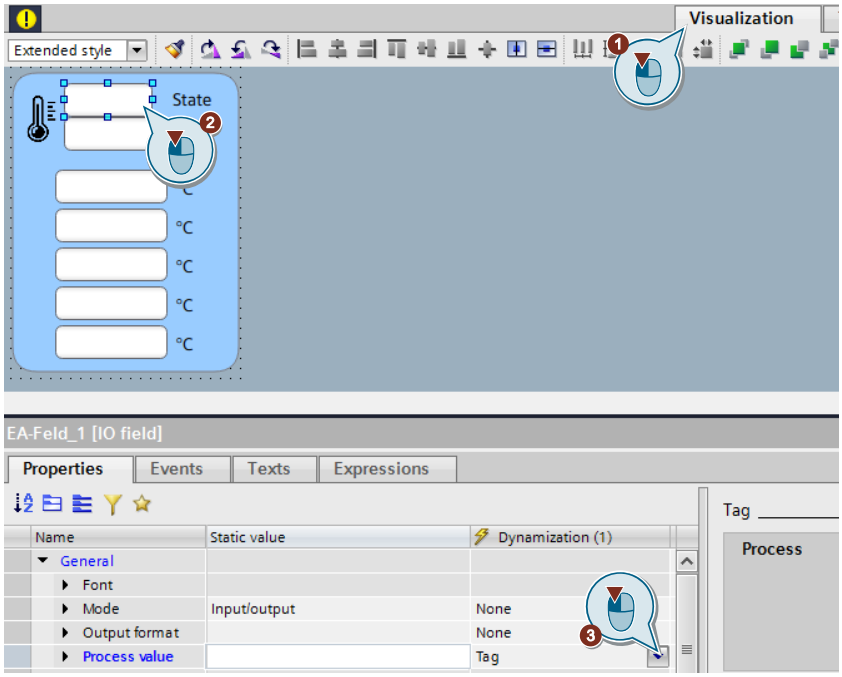
Abbildung 2-16



Verknüpfen von Visualisierung und Schnittstelle

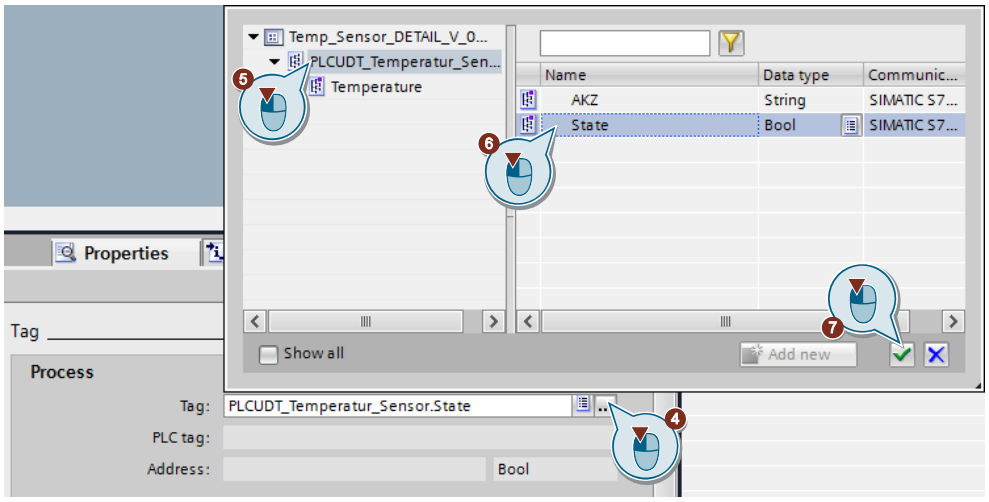
1. Wechseln Sie zurück in den Reiter "Visualisierung".
2. Selektieren Sie das oberste EA-Feld.
3. Navigieren Sie in den Einstellungen zu Allgemein->Prozesswert und ändern Sie die Dynamisierung zu "Variable".

Abbildung 2-17



4. Öffnen Sie im Bereich "Variable -> Prozess -> Variable" das Auswahlfenster für die anzubindende Variable.
5. Navigieren Sie im linken Bereich zum Eintrag "PLCUDT_Temperatur_Sensor" und selektieren Sie diesen.
6. Selektieren Sie im rechten Bereich den Eintrag "State".
7. Bestätigen Sie die Auswahl mit einem Klick auf den Haken.

Abbildung 2-18



8. Wiederholen Sie die Schritte 2. – 7. Für alle weiteren EA-Felder.

   Nutzen Sie dafür die Variablen "AKZ", sowie "Temperature[0]" bis "Temperature[4]" (Letztere befinden sich in dem Überverzeichnis "Temperature").

   Geben Sie abschließend die Version des Faceplates frei und beenden Sie damit die Bearbeitung.

2.3.4.2 Inneres Faceplate "Temp_Sensor"

Anlegen des Faceplate-Typs

1. Öffnen Sie in der Bibliothek mit einem Doppelklick auf "Neuen Typ hinzufügen" den gleichnamigen Editor, um ein neues Faceplate hinzuzufügen.
2. Wählen Sie im Dialog "Neuen Typ hinzufügen" den Typ "HMI Bildbaustein" aus.
3. Ändern Sie den Namen zu "Temp_Sensor".
4. Wählen Sie als Gerät "Unified Comfort Panels / WinCC Unified PC" aus.
5. Wählen Sie für die niedrigste Geräteversion "18.0.0.0" aus.
6. Bestätigen Sie die Auswahl mit einem Klick auf "OK". Das Faceplate wird erzeugt und im Editor geöffnet.

Anpassen der Visualisierung

Ändern Sie in den Eigenschaften des Faceplates die Größe auf folgende Werte:

- Größe – Breite: 200
- Größe – Höhe: 50

Fügen Sie nun die in Tabelle 2-4 aufgelisteten Basisobjekte hinzu und projektieren Sie damit die folgende Visualisierung:

Abbildung 2-19

Tabelle 2-4

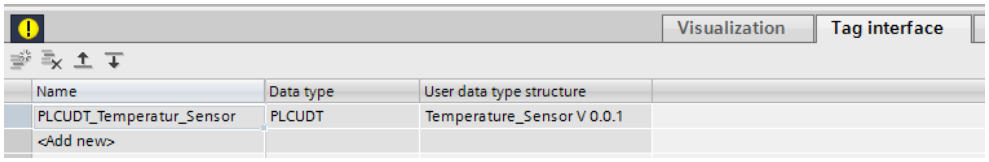
Basisobjekt	Anzahl	Anmerkungen
Grafikanzeige	1	Grafik: "Thermometer_Graphic V 0.0.1" (In Bibliothek des Anwendungsbeispiels enthalten)
EA-Feld	1	Modus: Ausgabe
Textfeld	1	-
Schaltfläche	1	Grafik: "Gear_Graphic V 0.0.1" (In Bibliothek des Anwendungsbeispiels enthalten)

Konfigurieren der Variablen Schnittstelle

Wechseln Sie in den Reiter "Variablen Schnittstelle", legen Sie mit einem Klick auf "<Hinzufügen>" eine neue Schnittstelle an und vergeben Sie folgende Parameter:

- Name: "PLCUDT_Temperatur_Sensor"
- Datentyp: "PLCUDT"
- Anwenderdatentyp-Struktur: "Temperature_Sensor V 0.0.1"

Abbildung 2-20



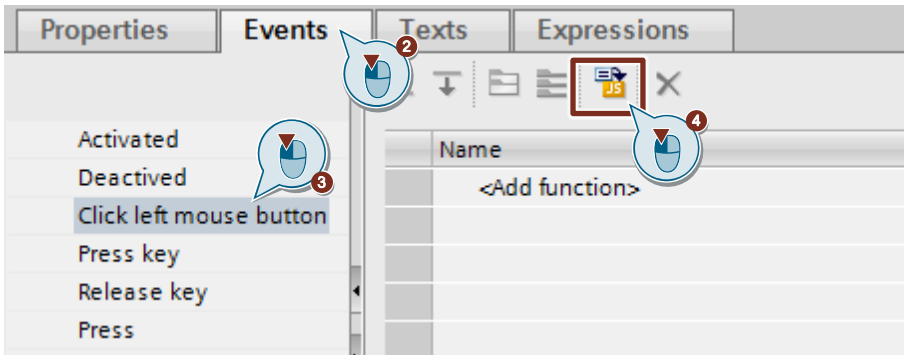
Verknüpfen von Visualisierung und Schnittstelle

1. Wechseln Sie zurück in den Reiter "Visualisierung".
2. Selektieren Sie das EA-Feld.
3. Navigieren Sie in den Einstellungen zu Allgemein > Prozesswert und ändern Sie die Dynamisierung zu "Variable".
4. Öffnen Sie im Bereich "Variable -> Prozess -> Variable" das Auswahlfenster für die anzubindende Variable.
5. Navigieren Sie im linken Bereich zum Eintrag "Temperature" und selektieren Sie diesen.
6. Selektieren Sie im rechten Bereich den Eintrag "[0]".
7. Bestätigen Sie die Auswahl mit einem Klick auf den Haken.

Einrichten des PopUp Faceplates

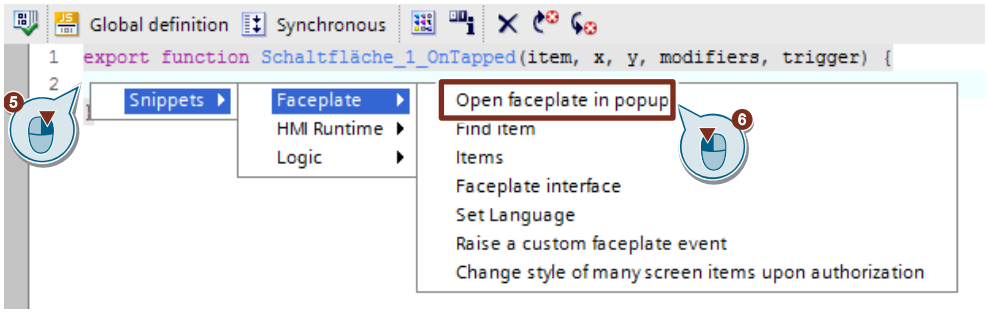
1. Selektieren Sie die Schaltfläche.
2. Wechseln Sie in den Reiter "Ereignisse".
3. Selektieren Sie den Eintrag "Linke Maustaste klicken".
4. Wandeln Sie die Funktion mit der entsprechenden Schaltfläche in ein Skript um.

Abbildung 2-21



5. Öffnen Sie das Auswahlmenü für Code Snippets mit einem Rechtsklick in die leere Zeile ("2").
6. Navigieren Sie zu "Snippets -> Faceplate -> Open faceplate in popup" und wählen Sie das Snippet mit einem Linksklick aus.

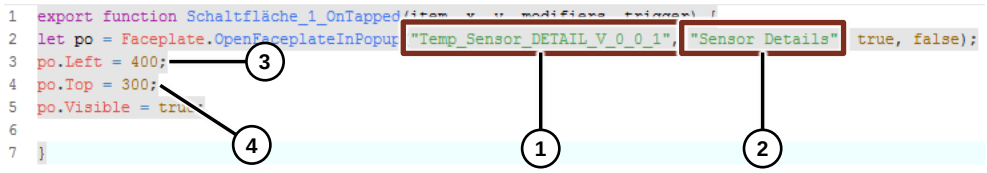
Abbildung 2-22



Passen Sie nun die Parameter des Snippets wie folgt an:

- Faceplate Type: "Temp_Sensor_DETAIL_V_0_0_1" (1)
- Title: "Sensor Details" (2)
- po.Left: 400 (3)
- po.Top: 300 (4)

Abbildung 2-23



Geben Sie abschließend die Version des Faceplates frei und beenden Sie damit die Bearbeitung.

Hinweis

Weitere Informationen zur Verwendung und Projektierung von Faceplates als PopUps finden Sie in Kapitel 3.5 Faceplates als PopUp

2.3.4.3 Haupt-Faceplate "Motor"

Anlegen des Faceplate-Typs

1. Öffnen Sie in der Bibliothek mit einem Doppelklick auf "Neuen Typ hinzufügen" den gleichnamigen Editor, um ein neues Faceplate hinzuzufügen.
2. Wählen Sie im Dialog "Neuen Typ hinzufügen" den Typ "HMI Bildbaustein" aus.
3. Ändern Sie den Namen zu "Motor".
4. Wählen Sie als Gerät "Unified Comfort Panels / WinCC Unified PC" aus.
5. Wählen Sie für die niedrigste Geräteversion "18.0.0.0" aus.

6. Bestätigen Sie die Auswahl mit einem Klick auf "OK". Das Faceplate wird erzeugt und im Editor geöffnet.

Anpassen der Visualisierung

Ändern Sie in den Eigenschaften des Faceplates die Größe auf folgende Werte:

- Größe – Breite: 250
- Größe – Höhe: 250

Fügen Sie nun die in Tabelle 2-5 aufgelisteten Basisobjekte hinzu und projektieren Sie damit die folgende Visualisierung:

Abbildung 2-24

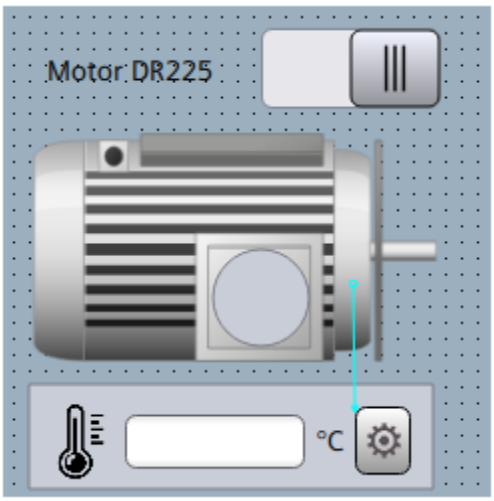


Tabelle 2-5

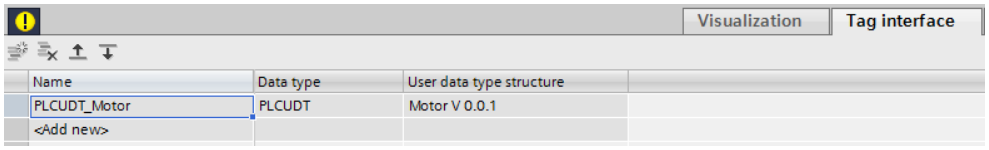
Basisobjekt	Anzahl	Anmerkungen
Grafikanzeige	1	Grafik: "Motor_Graphic V 0.0.1" (In Bibliothek des Anwendungsbeispiels enthalten)
Kreis	1	-
Textfeld	1	-
Schalter	1	-
Linie	1	Farbe: 0; 255; 255
Faceplate-Container	1	Hinzufügen via Drag & Drop des Faceplate-Typs "Temp_Sensor" aus der Bibliothek

Konfigurieren der Variablen Schnittstelle

Wechseln Sie in den Reiter "Variablen Schnittstelle", legen Sie mit einem Klick auf "<Hinzufügen>" eine neue Schnittstelle an und vergeben Sie folgende Parameter:

- Name: "PLCUDT_Motor"
- Datentyp: "PLCUDT"
- Anwenderdatentyp-Struktur: "Motor V 0.0.1"

Abbildung 2-25



Verknüpfen von Visualisierung und Schnittstelle

Wechseln Sie zurück in den Reiter "Visualisierung".

Schalter

1. Selektieren Sie den Schalter.
2. Navigieren Sie in den Einstellungen zu "Allgemein -> Zustand Schalter" und ändern Sie die Dynamisierung zu "Variable".
3. Öffnen Sie im Bereich "Variable -> Prozess -> Variable" das Auswahlfenster für die anzubindende Variable.
4. Navigieren Sie im linken Bereich zum Eintrag "PLCUDT_Motor" und selektieren Sie diesen.
5. Selektieren Sie im rechten Bereich den Eintrag "State".
6. Bestätigen Sie die Auswahl mit einem Klick auf den Haken.

Kreis

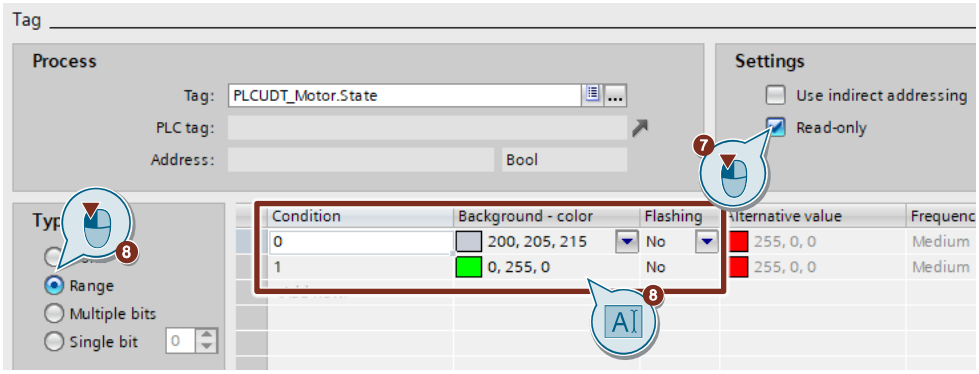
1. Selektieren Sie den Kreis.
2. Navigieren Sie in den Einstellungen zu "Gestaltung -> Hintergrund – Farbe" und ändern Sie die Dynamisierung zu "Variable".
3. Öffnen Sie im Bereich "Variable -> Prozess -> Variable" das Auswahlfenster für die anzubindende Variable.
4. Navigieren Sie im linken Bereich zum Eintrag "PLCUDT_Motor" und selektieren Sie diesen.
5. Selektieren Sie im rechten Bereich den Eintrag "State".
6. Bestätigen Sie die Auswahl mit einem Klick auf den Haken.

7. Setzen Sie im Bereich "Variable -> Prozess -> Einstellungen" den Haken bei "Nur Lesen".
8. Wählen Sie im Bereich "Variable -> Prozess -> Typ" den Typ "Bereich" und vergeben Sie folgende Parameter:

Tabelle 2-6

Bedingung	Hintergrund - Farbe	Blinken
0	200; 205; 215	Nein
1	0; 255; 0	Nein

Abbildung 2-26



Geben Sie abschließend die Version des Faceplates frei und beenden Sie damit die Bearbeitung.

Ergebnis

Sie haben drei Faceplates, sowie die zugehörigen Anwenderdatentypen angelegt und miteinander verknüpft. Sie können nun das Faceplate "Motor" in einem Bild instanziieren und über dessen Schnittstelle alle Informationen (Variablen), die auch für die tieferliegenden Faceplates benötigt werden, anbinden.

3 Engineering

3.1 Der Datentyp "Konfigurations-String"

Der Datentyp "Konfigurations-String" gibt Ihnen die Möglichkeit einen Freitext mit unbegrenzter Zeichenanzahl an ein Faceplate zu übergeben.

Damit können Sie beispielsweise einen Filter-Parameter an eine Meldeanzeige in einem Faceplate übergeben.

Beispiel

Im folgenden Beispiel soll ein vordefinierter Filter mit Hilfe einer Schaltfläche an eine Meldeanzeige, die sich in einem Faceplate befindet, übergeben werden. Der Filter wird in Form einer Zeichenkette an die entsprechende Eigenschaft übergeben und von der Meldeanzeige ausgewertet.

Der Filter soll nur Meldungen der Priorität 8 und mit einer höheren ID als 28 anzeigen. Die verwertbare Zeichenkette lautet "Priority = 8 AND ID > 28" (ohne Anführungszeichen)

Hinweis

Bei der zu übergebenden Zeichenkette muss auf eine korrekte Schreibweise geachtet werden, damit diese richtig von der Meldeanzeige ausgewertet werden kann. Zur Sicherstellung können Sie den Filter zuvor direkt in der Meldeanzeige anlegen und die Zeichenkette dort herauskopieren.

Vorbereitungen Faceplate

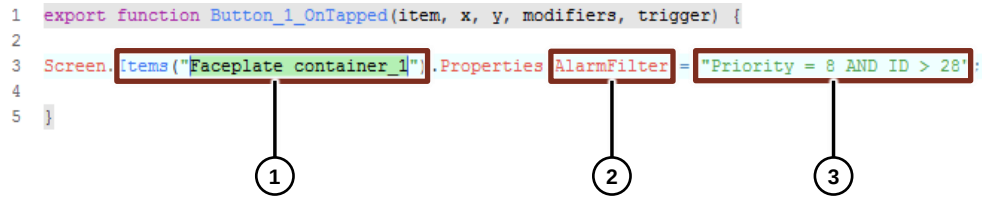
1. Erstellen Sie einen neuen Faceplate-Typ, der mindestens eine Meldeanzeige beinhaltet.
2. Legen Sie eine „Eigenschaften Schnittstelle“ vom Datentyp "Konfigurations-String" an.
3. Legen Sie für die Eigenschaft "Allgemein -> Filter" der Meldeanzeige eine Dynamisierung vom Typ „Eigenschaften Schnittstelle“ an und weisen Sie die zuvor erzeugte Konfigurations-String-Schnittstelle zu.
4. Geben Sie die Faceplate-Version frei.

Vorbereitungen Bild

1. Legen Sie eine Instanz des zuvor erstellten Faceplates in einem Bild an.
2. Fügen Sie dem Bild eine Schaltfläche hinzu.
3. Legen Sie für die Schaltfläche ein Klick-Ereignis an, wandeln Sie dieses in ein Skript um und geben Sie folgenden Code ein:

```
export function Button_1_OnTapped(item, x, y, modifiers, trigger) {
Screen.Items("Faceplate container_1").Properties.AlarmFilter =
"Priority = 8 AND ID > 28";
}
```

Abbildung 3-1



- Bildobjekt (1)
- Eigenschaft (2)
- Übergabeparameter (3)

Ergebnis

Wenn Sie zur Laufzeit auf die konfigurierte Schaltfläche klicken, wird der Filter an die Meldeanzeige übergeben und ausgewertet.

Gegenüberstellung "Mehrsprachiger Text"

Für die Übergabe von (mehrsprachigen) Texten an ein Faceplate nutzen Sie den Datentyp "Mehrsprachiger Text". Diesen können Sie an die Text-Eigenschaft jedes Bildobjekts anbinden, das diese unterstützt. Damit haben Sie die Möglichkeit den Bildobjekten jeder Instanz eines Faceplates verschiedene Texte zuzuweisen und somit den Mehraufwand einer Individualprojektierung zu vermeiden.

Alle Texte, die Sie über die "Mehrsprachiger Text"-Schnittstelle übergeben, werden auch in den Projekttexten aufgelistet und können dort editiert werden.

Der Datentyp "Konfigurations-String" ist nicht für die Übergabe an Text-Eigenschaften geeignet und nur für die Konfiguration bzw. Übergabe von Freitextparametern wie im obigen Beispiel beschrieben zu verwenden.

3.2 Übersetzung von Texten in Verbindung mit Faceplates

Möchten Sie die Texte von Faceplates in mehreren Sprachen verwenden, dann ist es nicht notwendig diese direkt in der Projektierungsphase zu übersetzen und einzubinden. Sie können die verwendeten Texte nachträglich in eine Excel-Datei exportieren, in dieser die Übersetzungen vornehmen und abschließend wieder in das Faceplate importieren. Bei diesem Prozess wird keine neue Version des Faceplates erzeugt.

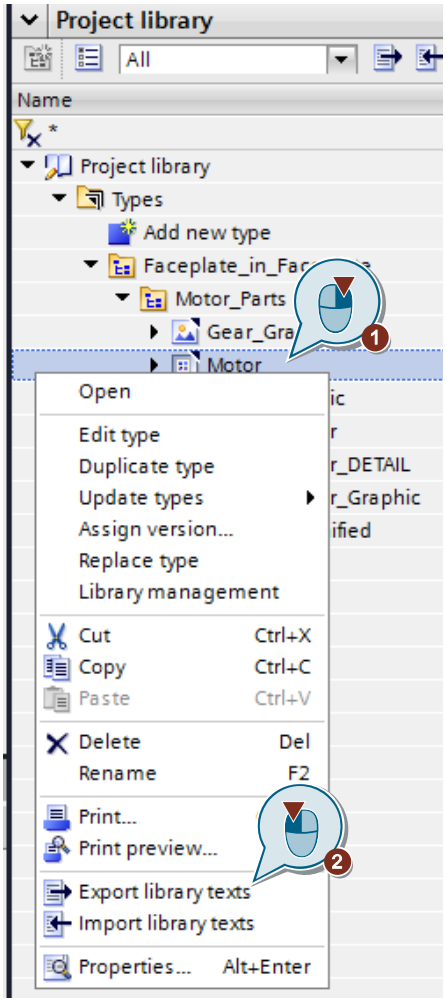
Hinweis

Es werden nur die Texte der als “Default“ gesetzten Version exportiert.
Stellen Sie diese ggf. zuvor auf die gewünschte Version um.

Beispiel Export

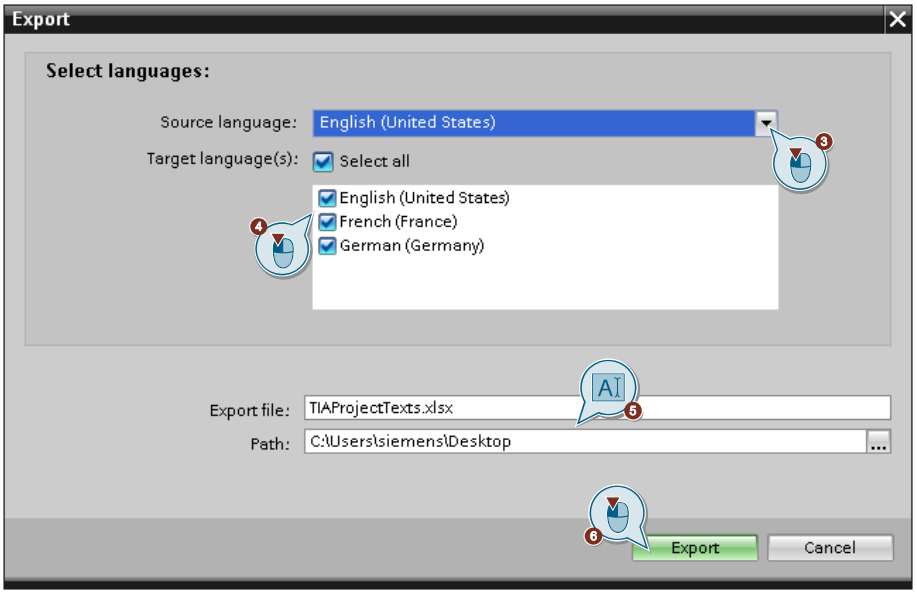
1. Stellen Sie sicher, dass die benötigten Sprachen sowohl unter “Sprachen & Ressourcen -> Projektsprachen“ als auch in den Runtime-Einstellungen unter “Sprache & Schriftart“ aktiviert sind.
2. Öffnen Sie mit einem Rechtsklick das Kontextmenü des entsprechenden Faceplate-Typs (Es muss der komplette Typ ausgewählt werden, keine Version) (1) und wählen Sie den Eintrag “Bibliothekstexte exportieren“ (2).

Abbildung 3-2



3. Wählen Sie im neu geöffneten Dialog die Quellsprache (3), sowie die zu exportierenden Sprachen (4) aus.
4. Geben Sie einen Namen für die Excel-Datei ein und wählen Sie einen Speicherort (5).
5. Bestätigen Sie den Export mit einem Klick auf “Exportieren“ (6).

Abbildung 3-3



6. Ein erfolgreicher Export wird Ihnen durch einen weiteren Dialog bestätigt. Diesen Dialog können Sie mit einem Klick auf “OK“ schließen.

Ergebnis

Sie haben erfolgreich die Texte eines Faceplates in eine Excel-Datei exportiert und können diese nun verwenden, um Übersetzungen vorzunehmen.

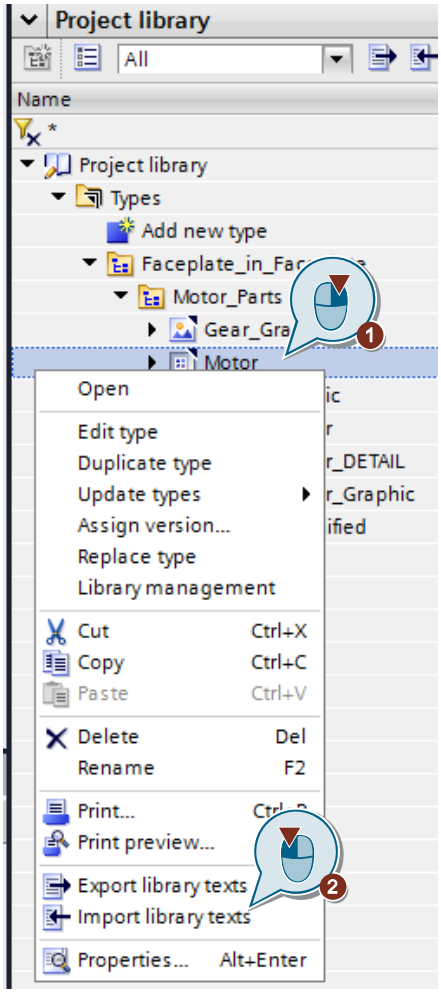
Beispiel Import

Voraussetzung:

Sie haben eine, zum entsprechenden Faceplate zugehörige, Excel Datei und haben in dieser Änderungen (Übersetzungen) vorgenommen.

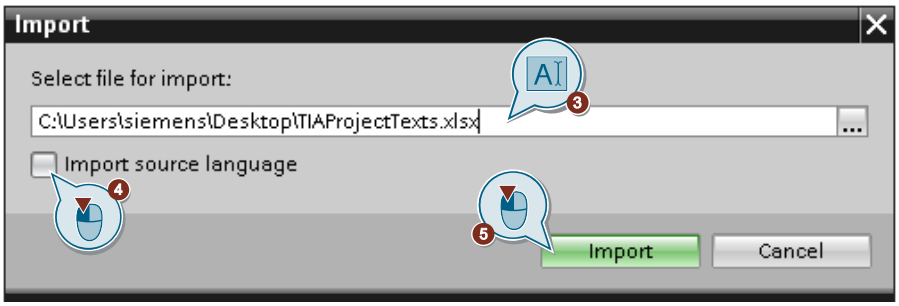
1. Öffnen Sie mit einem Rechtsklick das Kontextmenü des entsprechenden Faceplate-Typs (Es muss der komplette Typ ausgewählt werden, keine Version) (1) und wählen Sie den Eintrag “Bibliothekstexte importieren“ (2).

Abbildung 3-4



2. Wählen Sie im neu geöffneten Dialog den Pfad der zu importierenden Datei (3).
3. Aktivieren Sie bei Bedarf den Haken für die Option “Quellsprache importieren“ (4). Dies ist nur nötig, wenn an der Quellsprache auch Änderungen vorgenommen wurden.
4. Bestätigen Sie den Vorgang mit einem Klick auf “Importieren“ (5).

Abbildung 3-5



5. Ein erfolgreicher Import wird Ihnen durch einen weiteren Dialog bestätigt. Diesen Dialog können Sie mit einem Klick auf “OK“ schließen.

Ergebnis

Sie haben erfolgreich die (mehrsprachigen) Texte eines Faceplates aus einer Excel-Datei importiert und können diese nun im Faceplate verwenden.

3.3 Sichtbarkeit der Bildebenen in Faceplates umschalten

Sie haben die Möglichkeit, einzelne Bildebenen eines Faceplates sichtbar bzw. unsichtbar zu schalten. Dies kann hilfreich sein, wenn bestimmte Bildobjekte nicht immer notwendig sind oder nur in speziellen Fällen angezeigt werden sollen. Durch die Verwendung von Bildebenen und deren Sichtbarkeit ist es nicht nötig, dass Sie jedes Bildobjekt einzeln dynamisieren.

Sie können die Methode in jedem Ereignis projektieren, eine Umschaltung der Sichtbarkeit ist jedoch nur zur Laufzeit möglich. Sie können ein Faceplate also nicht mit bereits ausgeblendeten Bildebenen aufrufen.

Hinweis

Als möglichen Workaround können Sie die Umschaltung im Ereignis “Aufgebaut“ des Faceplates projektieren, wodurch der Effekt direkt beim Aufruf des Faceplates eintritt.

Weiterhin ist diese Funktion nur innerhalb des Faceplates verfügbar. Möchten Sie von außerhalb des Faceplates darauf zugreifen, müssen Sie die Übergabe der nötigen Daten/Informationen über eine Variablen- oder Eigenschaften Schnittstelle projektieren.

Für eine Zustandsabfrage verwenden Sie

```
Faceplate.Layers("Layer_X").Visible == true
```

oder

```
Faceplate.Layers("Layer_X").Visible == false
```

Möchten Sie einen Wert setzen, so verwenden Sie

```
Faceplate.Layers("Layer_X").Visible = true
```

oder

```
Faceplate.Layers("Layer_X").Visible = false
```

Hinweis

Ersetzen Sie den Platzhalter “X” im Ausdruck “Layer_X“ durch die entsprechende Nummer der Bildebene.

Sie können diese Funktion beliebig mit allen gängigen Logikfunktionen verknüpfen.

3.4 Austauschen der im Projekt verwendeten Version

Sie haben die Möglichkeit, die im Projekt zu verwendende Version eines Faceplates zu wechseln, um beispielsweise die Funktion verschiedener Stände zu testen.
Dabei können Sie in der Versionierung sowohl vor-, als auch zurückspringen.

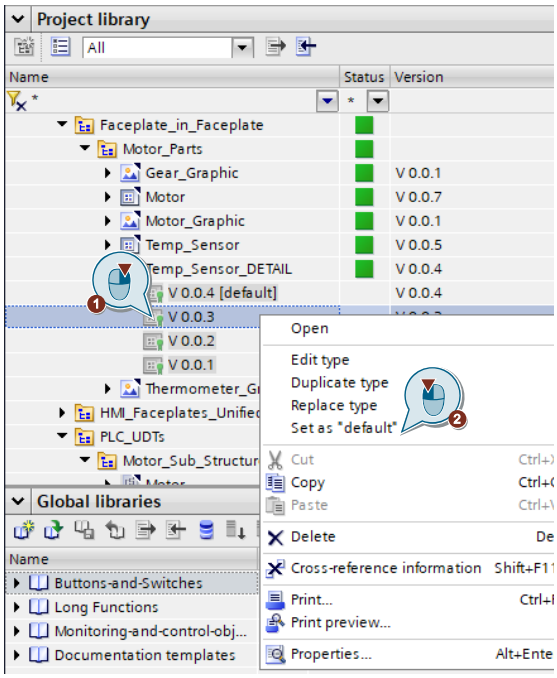
Die aktuell verwendete Version erkennen Sie daran, dass hinter der Versionsnummer der Ausdruck “ [Default]“ steht, z.B. “V 0.0.2 [Default]“.

Beispiel

In diesem Beispiel ändern Sie die Default Version eines Faceplates von V 0.0.4 zu V 0.0.3 und aktualisieren die im Projekt verwendete Instanz.

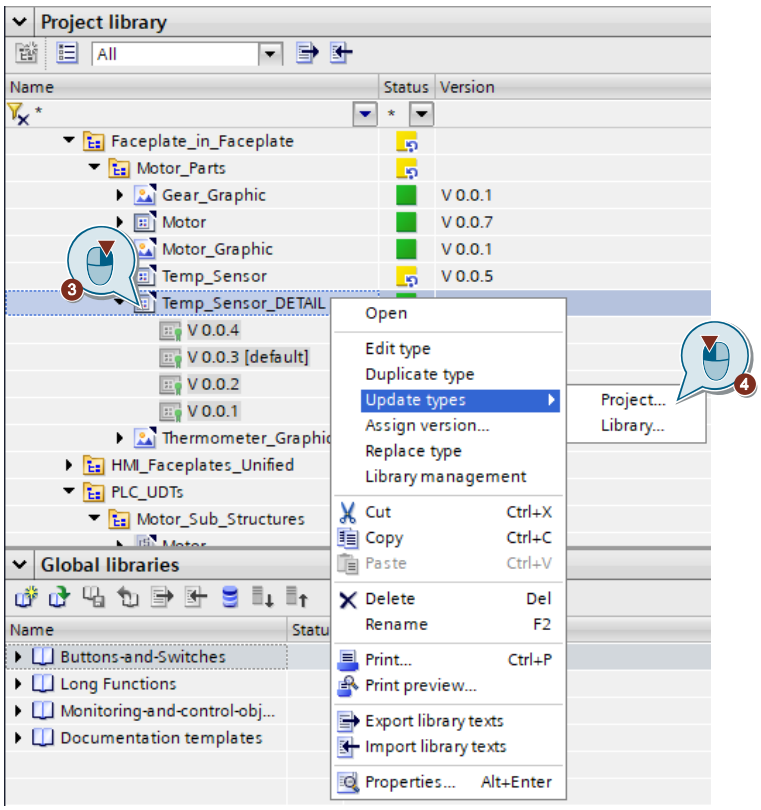
1. Selektieren Sie in der Baum-Struktur des Faceplates “ Temp_Sensor_DETAIL“ die Version V 0.0.3 und öffnen Sie mit der rechten Maustaste das Kontextmenü der Version (1).
2. Klicken Sie auf “Als “Default“ setzen“ (2). Das “[Default]“-Suffix wird zu dieser Version verschoben.

Abbildung 3-6



3. Öffnen Sie das Kontextmenü des Faceplate-Typs mit einem Rechtsklick (3) und wählen Sie den Eintrag “Typen aktualisieren > Projekt …“ (4).

Abbildung 3-7

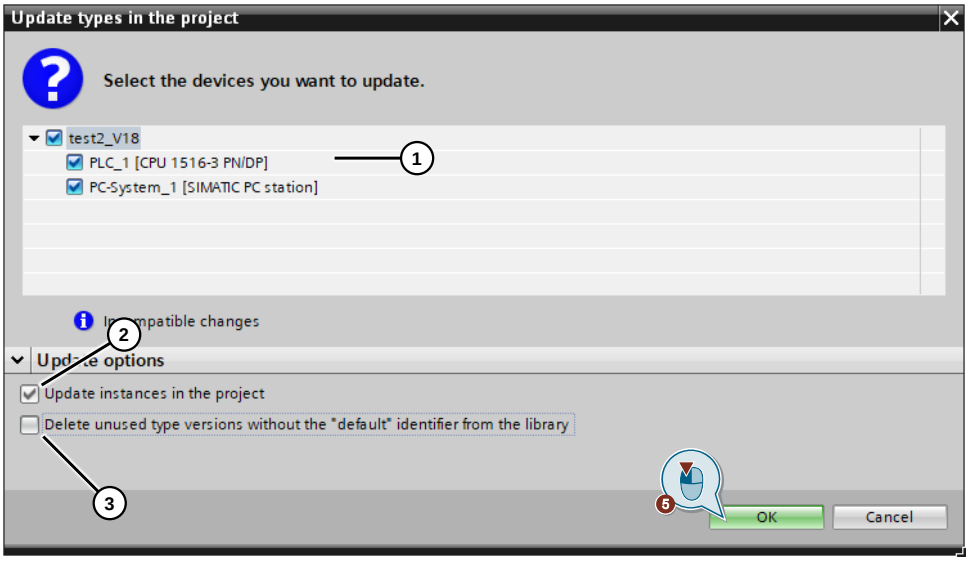


4. Ein Dialogfenster (“Typen im Projekt aktualisieren“) öffnet sich. Behalten Sie die Standardeinstellungen bei und bestätigen Sie die Auswahl mit einem Klick auf “OK“ (5).

Hinweis

Im Dialogfenster können Sie (sofern möglich) einstellen, auf welchen Geräten die Änderungen übernommen werden sollen (1), ob die in den Geräten verwendeten Instanzen aktualisiert werden sollen (2) und ob etwaige weitere Versionen des Faceplates, welche nicht als “Default“ markiert sind, aus der Bibliothek gelöscht werden sollen (3).

Abbildung 3-8



Ergebnis

Sie haben die im Projekt verwendete Version des Faceplates von V 0.0.4 zu V 0.0.3 geändert.

3.5 Teilen von Faceplates über globale Bibliotheken

Um bereits erstellte Faceplates mit anderen Benutzern oder in mehreren Projekten nutzen zu können und somit einen Mehraufwand durch erneutes Erstellen zu vermeiden, können Sie Faceplates mit Hilfe von globalen Bibliotheken verteilen.

Sie können bereits erstellte Faceplates aus dem Ordner “Typen“ der Projektbibliothek in den gleichnamigen Ordner einer globalen Bibliothek kopieren und diese speichern. Wenn Sie die globale Bibliothek zum Beispiel auf einem Netzlaufwerk speichern, können alle Benutzer mit Zugriff auf dieses Netzlaufwerk die Bibliothek und deren Inhalt jederzeit abrufen und bei Bedarf in die jeweilige lokale Projektbibliothek kopieren.

Sie können die globale Bibliothek auch lokal abspeichern und auf anderen Wegen mit anderen Personen teilen oder bei Bedarf mit einem anderen Projekt darauf zugreifen und die enthaltenen Inhalte (z.B. Faceplates) wiederverwenden.

3.6 Faceplates als PopUp

Faceplates können neben ihrer statischen Projektierung in einem Faceplate-Container auch als PopUp-Fenster konfiguriert werden. Dadurch verlieren Faceplates Ihre örtliche Bindung und können beispielsweise per Drag&Drop beliebig innerhalb der Runtime platziert werden, bei Bildwechseln im Vordergrund verbleiben, oder geschlossen werden, wenn Sie deren Anzeige nicht mehr benötigen.

Der Aufruf als PopUp-Fenster lässt sich sowohl aus einem Bild als auch aus einem weiteren Faceplate heraus projektieren. Dazu projektieren Sie ein Ereignis an einem Bildobjekt und verwenden das Code Snippet ”OpenFaceplateInPopup“.

Abhängig davon, ob Sie “OpenFaceplateInPopup“ in einem Bild oder in einem Faceplate verwenden, unterscheiden sich diese Code Snippets in einigen Punkten.

Übersicht

In diesem Kapitel erfahren Sie, wie Sie:

- Ein Faceplate als PopUp öffnen und über ein Skript verschalten → Kapitel 3.6.1
- Ein Faceplate im Bild projektieren und die Verschaltung in der Runtime ändern können → Kapitel 3.6.2
- Ein Faceplate aus einem Faceplate öffnen → Kapitel 3.6.3

Beispiel

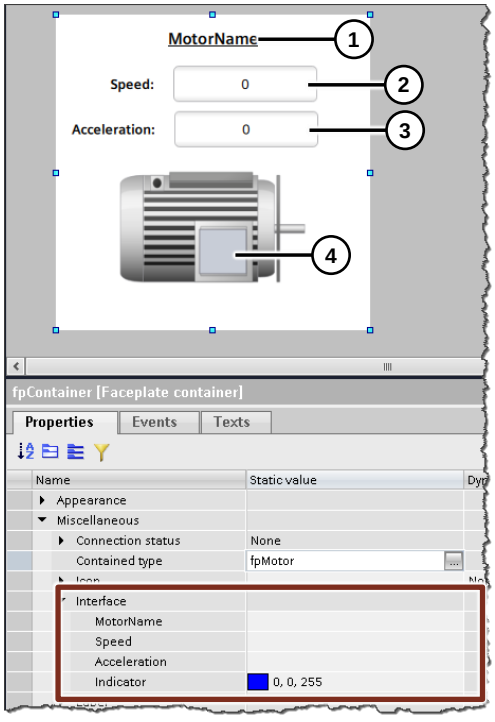
Hinweis

Die in den folgenden Beispielen verwendeten Elemente (Faceplates, Bilder, Variablen, …) dienen lediglich der Veranschaulichung und sind NICHT Teil des Beispielprojekts.

Zur besseren Verständlichkeit, wird nachfolgend die Verschaltung anhand des Faceplates "fpMotor" erklärt. Dieses besitzt:

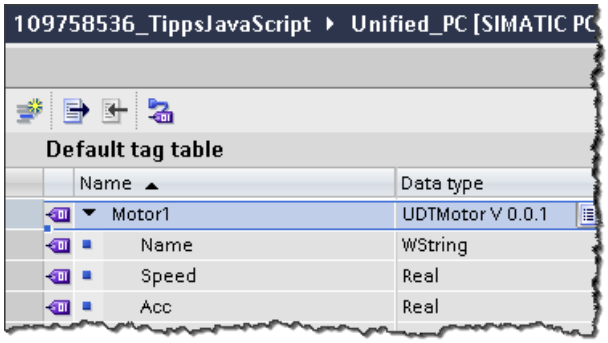
- drei Variablen Schnittstellen ("MotorName" (1), "Speed" (2) und "Acceleration" (3)) und
- eine Eigenschaften Schnittstelle ("Indicator" (4)).

Abbildung 3-9



Dieses Faceplate soll als PopUp aufgerufen und mit dem UDT "UDTMotor" verschaltet werden.

Abbildung 3-10, "UDTMotor"



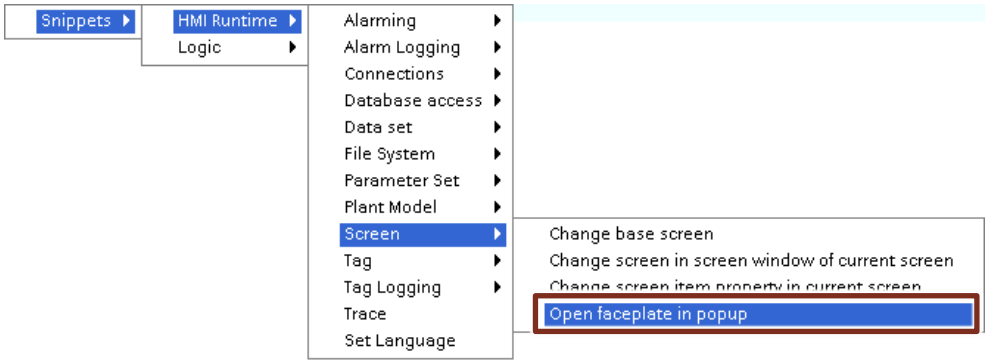
3.6.1 Faceplate als PopUp öffnen

In diesem Beispiel soll über eine Schaltfläche das Faceplate "fpMotor" als PopUp geöffnet werden und die Schnittstelle entsprechend verschaltet werden.

Code Snippet

Für den Aufruf von Faceplates gibt es das Snippet "Open faceplate in popup".

Abbildung 3-11

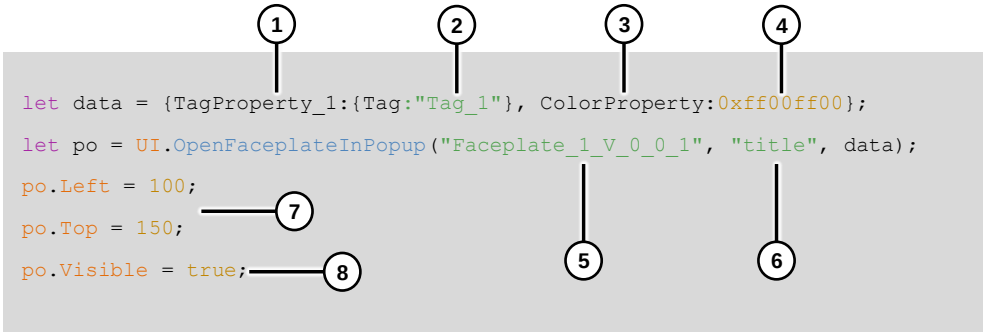



```
let data = {TagProperty_1:{Tag:"Tag_1"}, ColorProperty:0xff00ff00};
let po = UI.OpenFaceplateInPopup("Faceplate type_1", "title", data);
po.Left = 100;
po.Top = 150;
po.Visible = true;
```

Die Schnittstelle des Faceplates setzt sich aus folgenden Teilen zusammen:

1. der Schnittstellen Variablenname des Faceplates.
2. der Variablenname der HMI-Variable die übergeben werden soll.
3. der Name der Schnittstellen-Eigenschaft.
4. der übergebene Wert der Eigenschaft.
5. der Faceplate-Bezeichnung.
6. dem Titel des PopUp Fensters.
7. der Position des PopUp Fensters.
8. der Sichtbarkeit des PopUp Fensters.

Abbildung 3-12



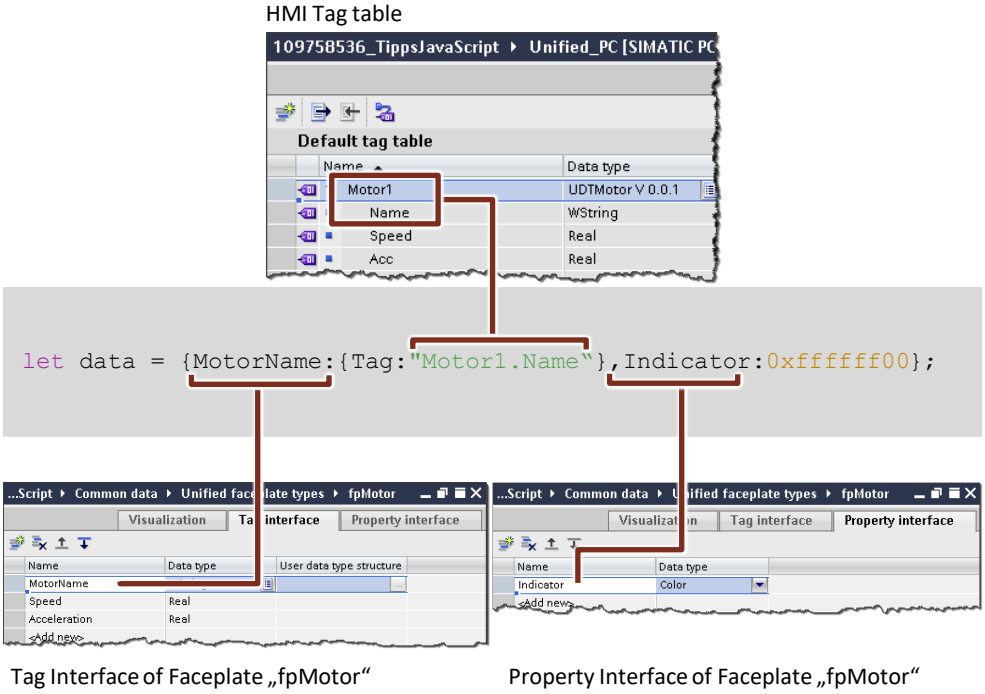
Hinweis

Weitere Informationen zu den Schnittstellenparametern sowie zu den optionalen Parametern "parentScreen" und "Visibility" finden Sie im Handbuch unter der Methode "OpenFaceplateInPopup":

https://support.industry.siemens.com/cs/de/de/view/109813308/159974440971

Anhand des Beispiels "fpMotor" und des zu verschaltenden "UDTMotor" ergibt sich folgende Verschaltung:

Abbildung 3-13

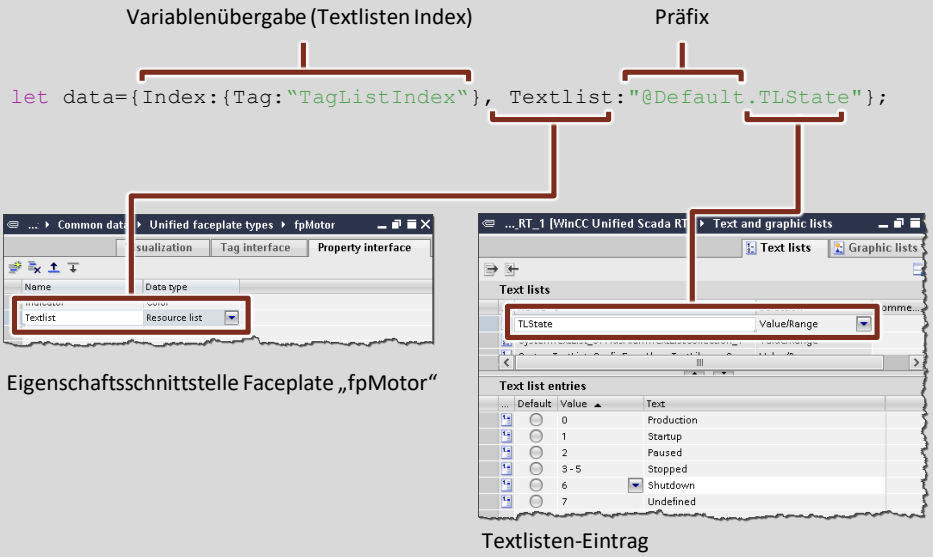


Hinweis

Wenn Sie eine Textliste über die Variablen Schnittstelle übergeben, benötigen Sie zusätzlich noch eine Variablenübergabe, um den Wert (Index) der Textliste zu übergeben.

Der Übergabeparameter für eine Textliste selbst setzt sich aus den beiden folgenden Angaben zusammen:

- Bezeichnung der Eigenschaften Schnittstelle (hier "Textlist")
- Präfix ("@Default.") + Name der Textliste (hier: "TLState")



Eigenschaftsschnittstelle Faceplate „fpMotor“

Textlisten-Eintrag

Die Übergabe von Grafiklisten wird aktuell nicht unterstützt.

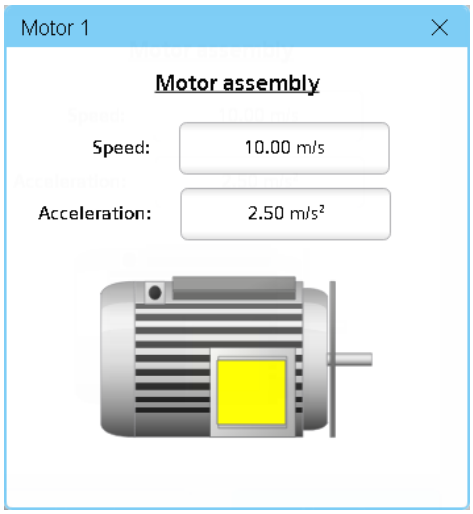
Ergebnis

Die nachfolgende Übersicht zeigt das fertig konfigurierte Code-Snippet.

```
let data={MotorName:{Tag:"Motor1.Name"},Speed:{Tag:"Motor1.Speed"},
Acceleration:{Tag:"Motor1.Acc"},Indicator:0xffffff00};
let po = UI.OpenFaceplateInPopup("fpMotor", "Motor 1", data);
po.Left = 100;
po.Top = 150;
po.Visible = true;
```

Nachdem die Schaltfläche in der Runtime betätigt wurde, erscheint das Faceplate als PopUp mit den übergebenen Daten.

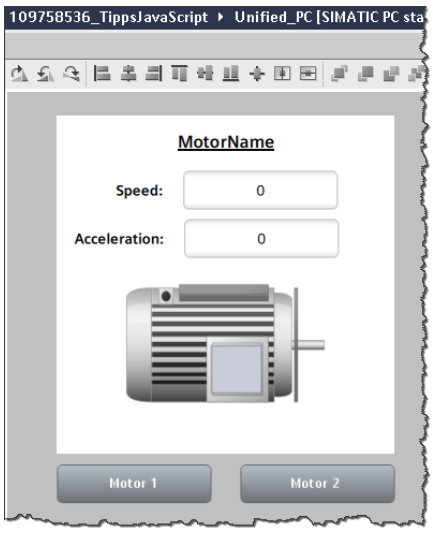
Abbildung 3-14



3.6.2 Faceplate-Verschaltung im Bild ändern

Im zweiten Anwendungsfall soll das Faceplate "fpMotor" im Bild projektiert werden. Über zwei Schaltflächen soll ein Wechsel der Schnittstelle in der Runtime von UDT Motor 1 auf UDT Motor 2 ermöglicht werden.

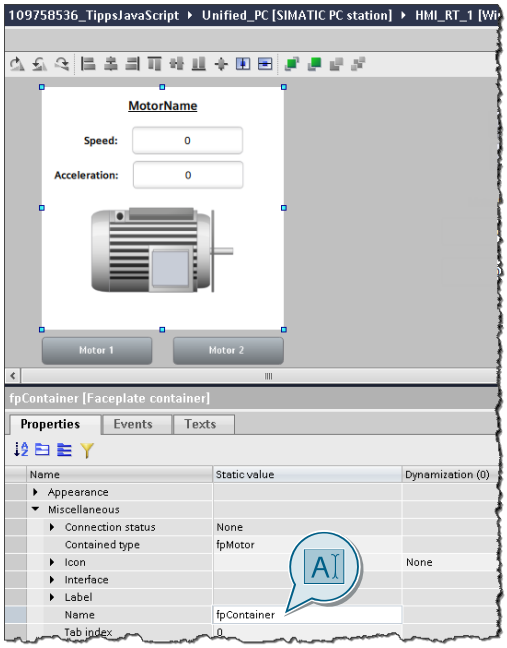
Abbildung 3-15



Projektierung

1. Platzieren Sie das Faceplate "fpMotor" per Drag&Drop im Bild.
2. Benennen Sie den automatisch erstellten Faceplate Container in "fpContainer" um.
3. Fügen Sie zwei Schaltflächen mit der Beschriftung "Motor 1" bzw. "Motor 2" hinzu.

Abbildung 3-16



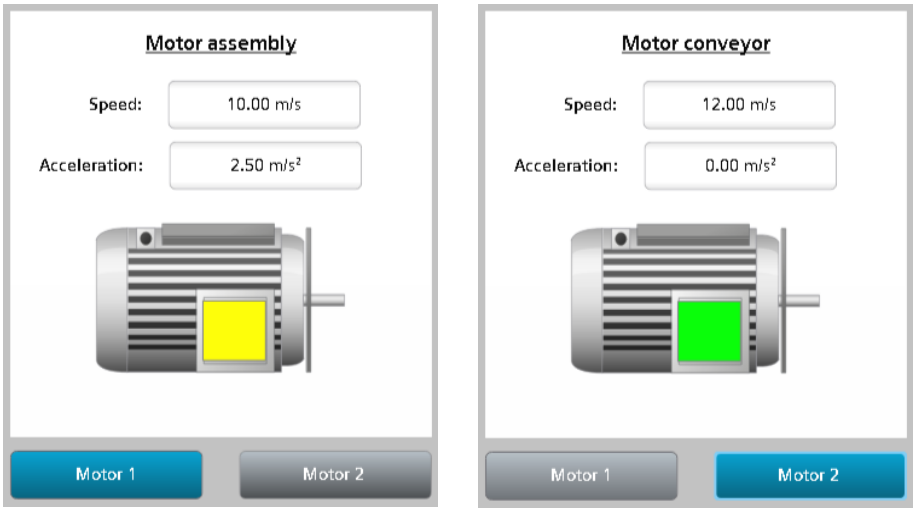
4. Fügen Sie dem Ereignis "Linke Maustaste klicken" der Schaltfläche für Motor 1 folgendes Skript hinzu.

```
Screen.Items("fpContainer").Properties.MotorName.Tag = "Motor1.Name";
Screen.Items("fpContainer").Properties.Speed.Tag = "Motor1.Speed";
Screen.Items("fpContainer").Properties.Acceleration.Tag = "Motor1.Acc";
Screen.Items("fpContainer").Properties.Indicator = 0xff00ff08;
```

5. Wiederholen Sie Schritt 4 an der Schaltfläche für Motor 2 und ersetzen Sie dabei "Motor1" durch "Motor2".

Ergebnis in der Runtime

Abbildung 3-17



3.6.3 Faceplate aus Faceplate öffnen

In WinCC Unified haben Sie zusätzlich die Möglichkeit aus einem bereits geöffneten Faceplate ein weiteres Faceplate zu öffnen.

Code Snippet

Innerhalb eines Faceplates haben Sie ebenfalls die Möglichkeit, über das Snippet "Open faceplate in popup", ein weiteres Faceplate zu öffnen.

Abbildung 3-18


```
let po = Faceplate.OpenFaceplateInPopup("Faceplate type_1", "title",
true, false);
po.Left = 100;
po.Top = 150;
po.Visible = true;
```

Hinweis

Wenn Sie aus einem Faceplate (z. B. "Faceplate_1") ein weiteres Faceplate (z. B. " Faceplate_2") aufrufen bzw. öffnen, werden die verschalteten Schnittstellen-Variablen automatisch an das zu öffnende Faceplate ("Faceplate_2") übertragen ("vererbt"). Hierfür sind keine zusätzlichen Projektierungsschritte erforderlich.

3.6.4 Faceplate schließen

Eine Möglichkeit ein Faceplate zu schließen, stellt die Methode "Faceplate.Close()" dar.

Legen Sie dafür innerhalb des entsprechenden Faceplates eine Schaltfläche an.

Fügen Sie ein Ereignis an der Schaltfläche hinzu (z. B. "Linke Maustaste klicken"), wandeln Sie das Ereignis in ein Skript um und fügen Sie folgenden Code ein:

```
Faceplate.Close();
```

Die Methode unterscheidet selbstständig zwischen zwei Anwendungsfällen:

- Handelt es sich bei dem Faceplate um einen (statischen) Faceplate-Container in einem Bild, so wird dieser unsichtbar geschaltet.
- Handelt es sich bei dem Faceplate um ein PopUp, so wird dieses geschlossen.

Hinweis

Die Methode Faceplate.Close() funktioniert nur innerhalb eines Faceplates. Ein Zugriff von außen ist damit nicht möglich. Sie muss daher bereits im Faceplate-Editor projektiert werden.

3.6.5 Der Invisible-Parameter

Der Parameter “Invisible“ steht an letzter Stelle der Methode “OpenFaceplateInPopup“ und ist optional. Der Parameter ist sowohl in “UI.OpenFaceplateInPopup(…)“ als auch in “Faceplate.OpenFaceplateInPopup(…)“ vorhanden. Der Standardwert des Parameters beim Einfügen des Snippets ist “false“ (invisible = false -> PopUp ist sichtbar). Wenn Sie den Wert zu “true“ ändern, ist das PopUp beim Öffnen unsichtbar.

Hinweis

Wenn Sie das Snippet “OpenFaceplateInPopup“ einfügen, wird gleichzeitig der Aufruf “po.Visible = true;“ eingefügt.
Dieser steht in direktem Gegensatz zum Invisible-Parameter (Visible <> Invisible) und hat immer Vorrang. Unabhängig davon, welchen Wert Sie für den Invisible-Parameter wählen, wird immer der Zustand übernommen, welcher in „po.Visible“ angegeben ist.
Möchten Sie dies vermeiden und nur den Invisible-Parameter verwenden, dann müssen Sie die Zeile mit dem entsprechenden Aufruf auskommentieren oder löschen.

Abbildung 3-19

```
let data = {TagProperty_1:{Tag:"Tag_1"}, ColorProperty:0xff00ff00};
let po = UI.OpenFaceplateInPopup("Faceplate_1_V_0_0_1", "title", data);
po.Left = 100;
po.Top = 150;
po.Visible = true;
```

```
let po = Faceplate.OpenFaceplateInPopup("Faceplate_1_V_0_0_1", "title", true, false);
po.Left = 100;
po.Top = 150;
po.Visible = true;
```

3.6.6 Der Parameter “IndependentWindow“

Die Methode „OpenFaceplateInPopup“ beinhaltet den boolschen Parameter „IndependentWindow“.

Abbildung 3-20

```
2  let po = Faceplate.OpenFaceplateInPopup("Faceplate_1_V_0_0_1", "title", true, false);
3  po.Left = 100;
4  po.Top = 150;
5  po.Visible = true;
```

```
Object/HmiPopupScreenWindow OpenFaceplateInPopup(String/HmiFaceplateType faceplateType, String title, Bool independentWindow, Bool invisible)
Opens the faceplate in a new Popup Window

faceplateType:
title:
independentWindow: (optional) When 'true' the lifetime of the PopUp is decoupled from the lifetime of the calling Faceplate
invisible: (optional) Set this parameter to 'true' to create the Faceplate invisible
```

Mit diesem Parameter können Sie, abhängig von weiteren Gegebenheiten, die Position und die sogenannte Lebensdauer eines PopUp-Faceplates beeinflussen.

In diesem Kapitel finden Sie eine Auflistung der möglichen Konfigurationen und deren Ergebnisse.

Hinweis

Der Parameter “IndependentWindow“ ist nur verfügbar, wenn Sie die “OpenFaceplateInPopup“-Methode in einem weiteren Faceplate aufrufen (Faceplate.OpenFaceplateInPopup(…)). Bei Verwendung in einem Bild (UI.OpenFaceplateInPopup(…)) unterscheidet sich der Aufbau der Methode und beinhaltet diesen Parameter nicht.

Abhängigkeit zum Aufrufenden Faceplate

- IndependentWindow = true
  - Das PopUp-Faceplate wird nicht geschlossen, wenn Sie das abhängige Faceplate schließen
- IndependentWindow = false
  - Wenn Sie das aufrufende Faceplate schließen, wird auch das PopUp-Faceplate geschlossen.

Verhalten bei Bildwechseln

- IndependentWindow = true
  - Das PopUp-Faceplate bleibt bei einem Bildwechsel (vordergründig) geöffnet
- IndependentWindow = false
  - Das PopUp-Faceplate wird bei einem Bildwechsel geschlossen (abgebaut)

Koordinatenursprünge

Der Koordinatenursprung (0/0) ist die Position, von welcher die Werte für “Oben“ und “Links“ gemessen werden (in Pixeln).

Dieser befindet sich, je nach Konfiguration, in der linken oberen Ecke

- des Hauptbilds.
- des Bildfensters.
- des Faceplates.
  (Handelt es sich um ein PopUp-Faceplate ist dessen Bildinhalt gemeint - ein möglicher Header wird hierfür nicht mit einbezogen).

3.6.7 Parentfunktion

Ein sogenanntes “Parent“ beschreibt ein Bildobjekt, das dem Bezugsobjekt direkt übergeordnet ist, wie beispielsweise

- das Bild, in dem sich ein Bildobjekt befindet.
- das Bildfenster, in dem sich ein Bild befindet.
- das Faceplate, in dem sich ein Bildobjekt befindet.
- das Bild, in dem sich ein Faceplate befindet bzw. dort aufgerufen wird.
- das Faceplate, aus dem ein PopUp-Faceplate aufgerufen wird

wobei immer das Erstgenannte Objekt das Parent-Objekt ist.

Der Zugriff auf ein Parent lässt sich über das Scripting projektieren und ermöglicht das Auslesen von Daten und Werten eines Parent-Objekts. Dies können, je nach Objekt, z.B. die Größe, die Position, die Farbe, der Name oder vergleichbare Daten sein. Die so ausgelesenen Informationen können dann im untergeordneten (abfragenden) Bildobjekt weiterverwendet werden.

Da das Projektieren mit „Parents“ sich von Fall zu Fall unterscheidet und es verschiedene Möglichkeiten gibt, dies umzusetzen, wird an dieser Stelle nicht weiter darauf eingegangen.

Nähere Informationen zu dem finden Sie in der TIA Portal Hilfe unter dem Stichwort „Parent“.

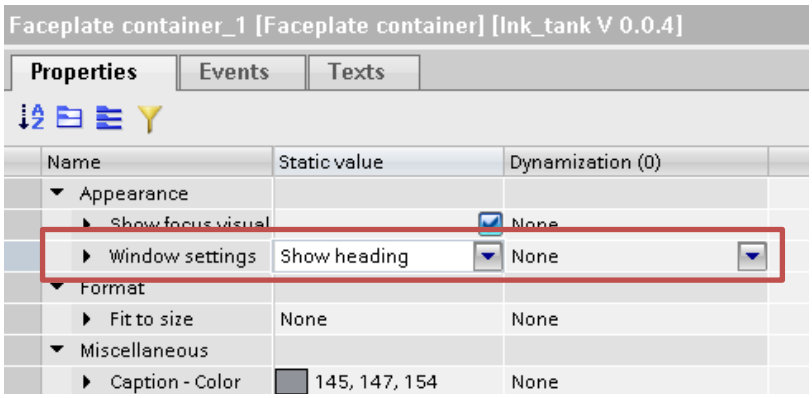
3.6.8 Header

Allgemeines

Bei der Projektierung eines Faceplates können Sie festlegen, ob dieses mit einem sogenannten “Header“ dargestellt werden soll (Faceplate-Container > Eigenschaften > Gestaltung > Fenstereinstellungen > Überschrift anzeigen).

Der Header kann beispielsweise den Titel des Faceplates und eine “Schließen“-Schaltfläche enthalten. Abhängig vom verwendeten Stil, ändern sich das Aussehen und die Höhe des Headers. Die Breite entspricht immer der des Faceplate-Containers (der Faceplate-Instanz).

Abbildung 3-21



Darstellungsverhalten bei Verwendung eines Headers

Wenn Sie den Header bei einer Faceplate-Instanz aktivieren, dann ergeben sich neue Verhaltensweisen, wenn Sie die Darstellung über die Eigenschaft “Größe Anpassen“ verändern:

- Größe Anpassen > Keine

  Mit dem Einblenden des Headers bleibt die ursprüngliche Größe des Faceplate-Containers bestehen. Durch die Anzeige des Headers verkleinert sich der Bereich für die Darstellung des Bildinhalts. Dadurch werden Bildlaufleisten eingeblendet.

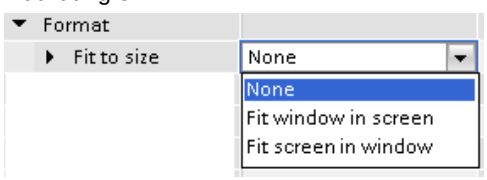
- Größe Anpassen > Bild an Fenster anpassen

  Der Bildinhalt des Faceplates wird gleichmäßig (Höhe und Breite) herunterskaliert, sodass der Header und der vollständige Bildinhalt - bei gleichbleibender Größe des Faceplate-Containers - angezeigt werden. Dies führt im Umkehrschluss zu einer vergrößerten “leeren Bildfläche“ in der Faceplate-Instanz.

- Größe Anpassen > Fenster an Bild anpassen

  Der Faceplate-Container wird automatisch vergrößert, sodass der Header und der vollständige Bildinhalt in Originalgröße angezeigt werden.

Abbildung 3-22



Einfluss des Headers auf den “IndependentWindow“-Parameter

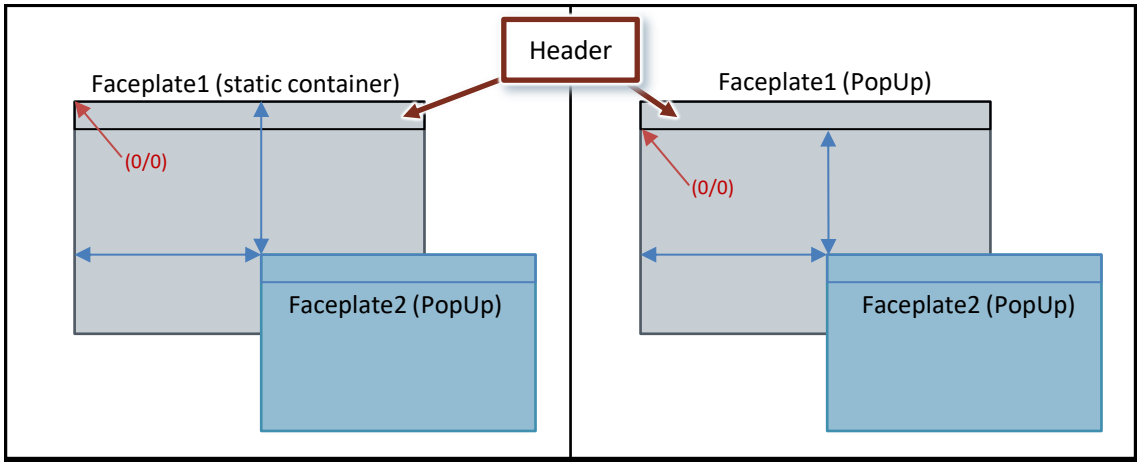
Die Verwendung eines Headers hat Einfluss auf das Verhalten des “IndependentWindow“-Parameters.

Rufen Sie ein PopUp-Faceplate aus einem weiteren Faceplate auf und nutzen den “IndependentWindow“-Parameter, dann verschiebt sich der Koordinatenursprung des PopUps.

Verwenden Sie einen Header in einem Faceplate,

- das in einem (statischen) Faceplate-Container in einem Bild liegt, dann wird bei Aufruf des untergeordneten PopUp-Faceplates die linke obere Ecke des Faceplate-Containers als Koordinatenursprung gewählt (Die mögliche existenz eines Headers wird ignoriert).
- das als PopUp-Faceplate (aus einem Bild heraus) geöffnet wurde, so wird bei Aufruf des untergeordneten PopUp-Faceplates die linke obere Ecke des Bildinhalts als Koordinatenursprung gewählt. Dies ist so realisiert, da ein Header eine variable Höhe haben kann (abhängig vom gewählten Stil). In diesem Fall müssen Sie ggf. die spezifische Höhe des Headers hinzurechnen, damit Sie die korrekten Werte für Position und Größe erhalten.

Abbildung 3-23



3.7 Fallbeispiele in Verbindung mit “IndependentWindow“-Parameter

Hinweis

In den folgenden Beispielen wird davon ausgegangen, dass Faceplate1 und Faceplate2 als PopUp-Faceplates konfiguriert sind, wobei Faceplate1 aus einem Bild und Faceplate2 aus Faceplate1 heraus aufgerufen wird. Außerdem ist bei beiden Faceplates ein Header aktiviert.

Tabelle 3-1

Nr.	Anwendungsfall	Kurzbeschreibung	Kapitel
1.	Faceplate2 soll Details einer Maschine darstellen. Dabei sollen die Informationen während der weiteren Bedienung der Visualisierung (z. B. Bildwechsel) einsehbar bleiben. Die Position des Faceplates wird absolut festgelegt, das Faceplate muss explizit geschlossen werden.	Position von Faceplate2 absolut in (Haupt-)Bild, Faceplate2 bleibt bei Bildwechsel oder Schließen von Faceplate1 geöffnet	3.7.1.1- Absolut und persistent (Bild, groß)
2.	Faceplate2 soll Details eines Motors darstellen. Dabei soll es sich neben der Ansicht der zugehörigen Maschine (Faceplate1) öffnen. Faceplate2 wird relativ zu Faceplate1 positioniert. Es soll beim Schließen von Faceplate1 oder einem Bildwechsel automatisch geschlossen werden. Geeignet für große Bilder / Bildschirme.	Position von Faceplate2 relativ zu Faceplate1, Faceplate2 wird bei Bildwechsel oder Schließen von Faceplate1 geschlossen	3.7.1.2 - Relativ und temporär (Bild, groß)
3.	Faceplate2 soll Details eines Motors darstellen. Dabei soll es sich neben der Ansicht der zugehörigen Maschine (Faceplate1) öffnen. Faceplate2 wird relativ zu Faceplate1 positioniert. Dabei sollen die Informationen während der weiteren Bedienung der Visualisierung (z. B. Bildwechsel) einsehbar bleiben. Die Position von Faceplate2 wird über die Parent-Funktion (relativ zu Faceplate1) festgelegt. So kann die Position beispielsweise über ein Skript kontrolliert und automatisch verändert werden. Dies ist besonders bei kleinen Bildern / Panels nützlich. Das Faceplate muss explizit geschlossen werden.	Position von Faceplate2 über Parent-Objekt ermittelt, Faceplate2 bleibt bei Bildwechsel oder Schließen von Faceplate1 geöffnet	3.7.1.3 - Relativ und persistent (Bild, klein)
4.	Wie 2., aber die Position von Faceplate2 wird über die Parent-Funktion (relativ zu Faceplate1) festgelegt. So kann die Position beispielsweise über ein Skript kontrolliert und automatisch verändert werden. Dies ist besonders bei kleinen Bildern / Panels nützlich.	Position von Faceplate2 über Parent-Objekt ermittelt, Faceplate2 wird bei Bildwechsel oder Schließen von Faceplate1 geschlossen	3.7.1.4- Relativ und temporär (Bild, klein)
5.	Wie 1., aber Faceplate1 wird in einem Bild innerhalb eines Bildfensters aufgerufen. Die Position beider Faceplates ist absolut zum Hauptbild festgelegt, unabhängig vom Bildfenster.	Position beider Faceplates unabhängig von Bildfenster, Faceplate2 bleibt bei Bildwechsel oder Schließen von Faceplate1 geöffnet	3.7.1.1- Absolut und persistent (Hauptbild)
6.	Wie 2., aber Faceplate1 wird in einem Bild innerhalb eines Bildfensters aufgerufen. Die Position von Faceplate1 ist absolut zum Hauptbild festgelegt	Position von Faceplate2 relativ zu Faceplate1, Faceplate1 unabhängig von Bildfenster, Faceplate2 wird bei Bildwechsel oder schließen von Faceplate1 geschlossen	3.7.2.2 - Relativ und temporär (Hauptbild)
7.	Wie 1., aber die Position von Faceplate1 wird über dessen Parent-Objekt festgelegt (Bild in Bildfenster). Die Position von Faceplate2 ist absolut zum Hauptbild festgelegt.	Position von Faceplate2 unabhängig von Bildfenster, Position von Faceplate1 relativ zu Bildfenster, Faceplate2 bleibt bei Bildwechsel oder Schließen von Faceplate1 geöffnet	3.7.2.3 - Absolut und persistent (Bildfenster)

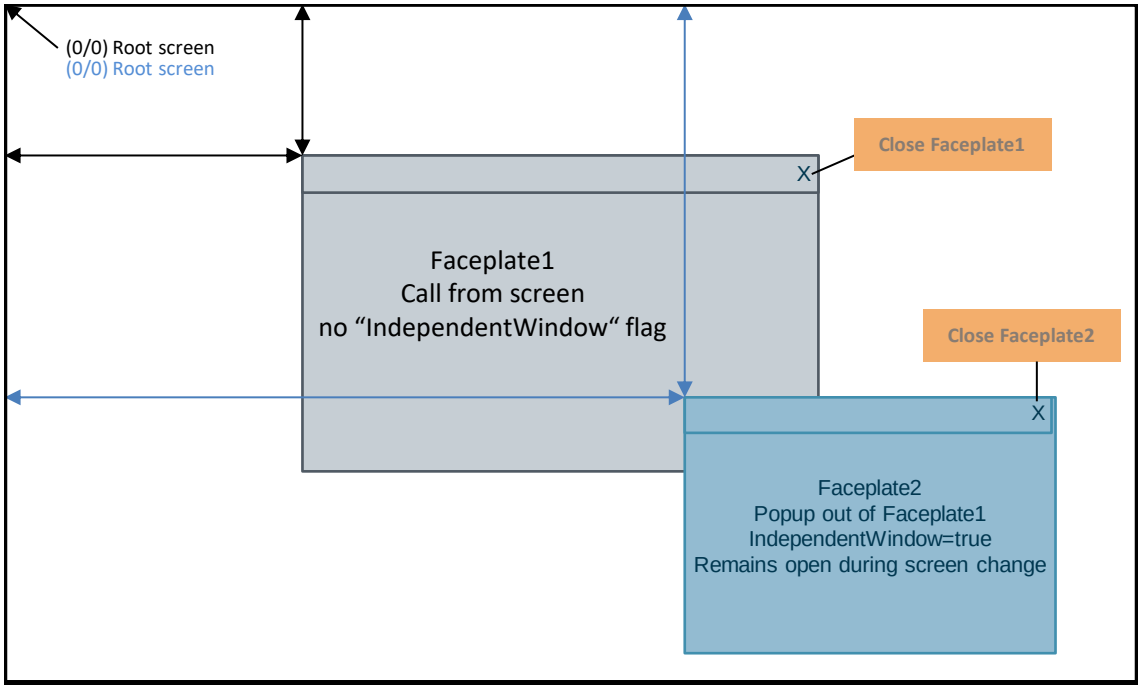
Nr.	Anwendungsfall	Kurzbeschreibung	Kapitel
8.	Wie 2., aber die Position von Faceplate1 wird über dessen Parent-Objekt festgelegt (Bild in Bildfenster).	Position von Faceplate2 relativ zu Faceplate1, Position von Faceplate1 relativ zu Bildfenster, Faceplate2 wird bei Bildwechsel oder Schließen von Faceplate1 geschlossen	3.7.2.4- Relativ und temporär (Bildfenster)

3.7.1 Aufruf in (Haupt-)Bild

3.7.1.1 Absolut und persistent (Bild, groß)

- Das aufrufende Faceplate1 wird in einem Bild aufgerufen.
- Das aufrufende Faceplate1 hat keinen “IndependentWindow“-Parameter.
- Das PopUp-Faceplate2 wird aus Faceplate1 aufgerufen.
- Der “IndependentWindow“-Parameter von Faceplate2 ist “true“.
- Der Koordinatenursprung (0/0) beider Faceplates entspricht dem des (Haupt-)Bilds.
- Faceplate1 und Faceplate2 lassen sich unabhängig voneinander schließen.
- Faceplate2 bleibt bei einem Bildwechsel (vordergründig) geöffnet.

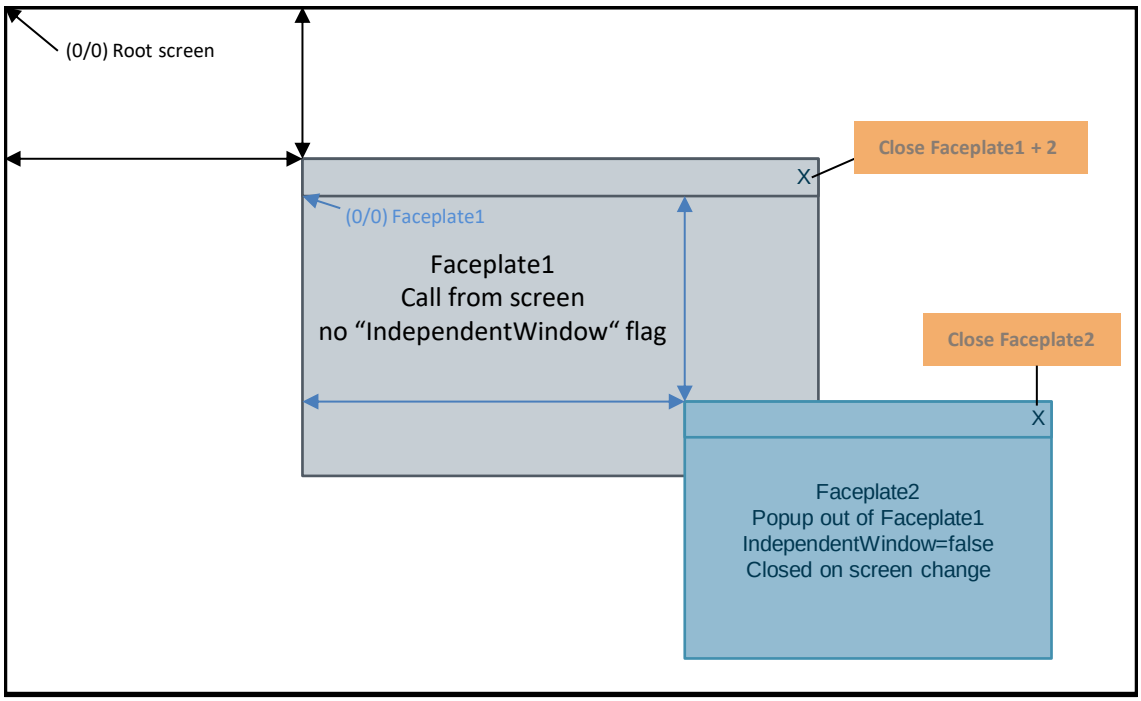
Abbildung 3-24



3.7.1.2 Relativ und temporär (Bild, groß)

- Das aufrufende Faceplate1 wird in einem Bild aufgerufen.
- Das aufrufende Faceplate1 hat keinen “IndependentWindow“-Parameter.
- Das PopUp Faceplate2 wird aus Faceplate1 aufgerufen.
- Der “IndependentWindow“-Parameter von Faceplate2 ist “false“.
- Der Koordinatenursprung (0/0) von Faceplate1 entspricht dem des (Haupt-) Bilds.
- Der Koordinatenursprung (0/0) von Faceplate2 entspricht dem des Bildinhalts von Faceplate1.
- Das Schließen von Faceplate1 schließt auch das Faceplate2.
- Faceplate2 wird bei einem Bildwechsel geschlossen.

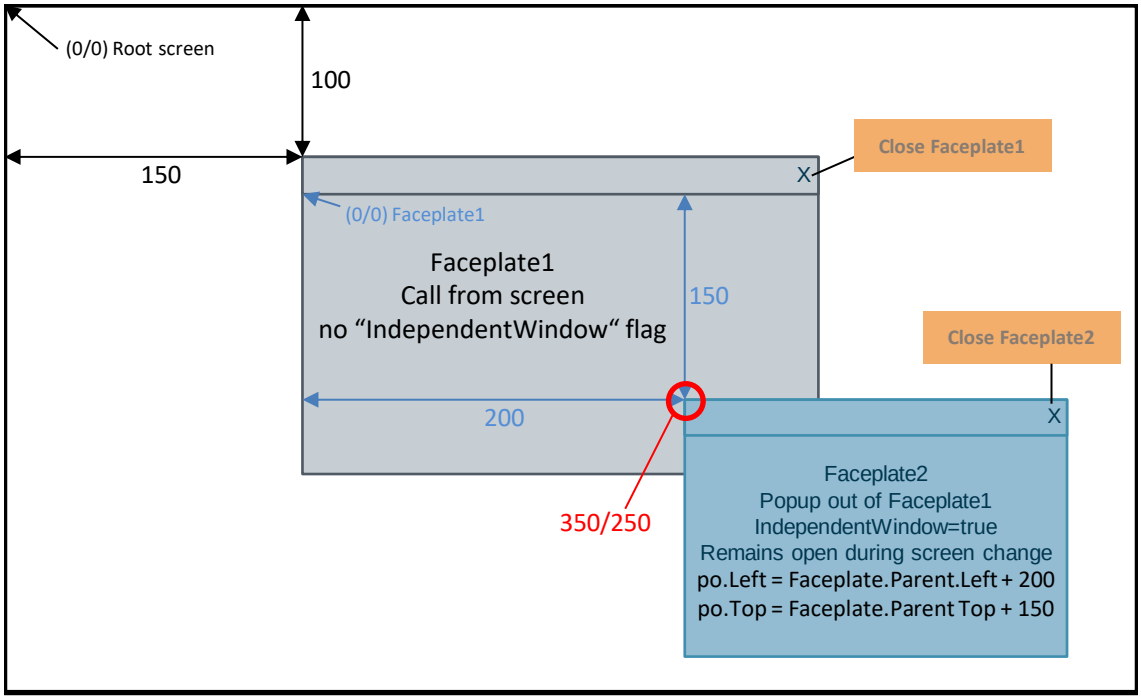
Abbildung 3-25



3.7.1.3 Relativ und persistent (Bild, klein)

- Das aufrufende Faceplate1 hat keinen “IndependentWindow“-Parameter.
- Das PopUp-Faceplate2 wird aus Faceplate1 aufgerufen.
- Der “IndependentWindow“-Parameter von Faceplate2 ist “true“.
- Der Koordinatenursprung (0/0) von Faceplate1 entspricht dem des (Haupt-) Bilds.
- Der Koordinatenursprung (0/0) von Faceplate2 entspricht dem des Bildinhalts von Faceplate1.
- In Faceplate2 wird auf dessen Parent-Objekt (Faceplate1) zugegriffen.
  - ⇨ Die Position von Faceplate1 wird ermittelt und dessen X und Y (Oben und Links) Koordinaten werden zu denen von Faceplate2 hinzuaddiert. Die Position von Faceplate2 wird also abhängig von der Position von Faceplate1 festgelegt.
- Faceplate1 und Faceplate2 lassen sich unabhängig voneinander schließen.
- Faceplate2 bleibt bei einem Bildwechsel (vordergründig) geöffnet.

Abbildung 3-26

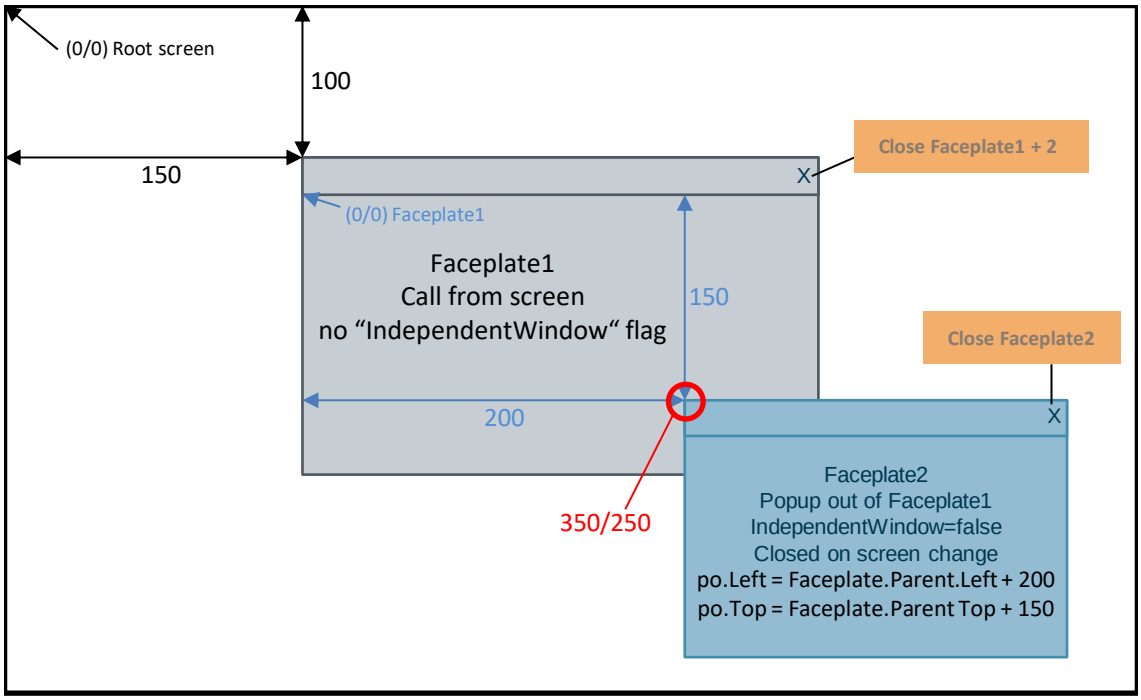


3.7.1.4 Relativ und temporär (Bild, klein)

- Das aufrufende Faceplate1 wird in einem Bild aufgerufen.
- Das aufrufende Faceplate1 hat keinen “IndependentWindow“-Parameter.
- Das PopUp-Faceplate2 wird aus Faceplate1 aufgerufen.
- Der “IndependentWindow“-Parameter von Faceplate2 ist “false“.
- Der Koordinatenursprung (0/0) von Faceplate1 entspricht dem des (Haupt) Bilds.
- Der Koordinatenursprung (0/0) von Faceplate2 entspricht dem des Bildinhalts von Faceplate1.
- In Faceplate2 wird auf dessen Parent-Objekt (Faceplate1) zugegriffen.
  - ⇨ Die Position von Faceplate2 wird relativ zur Position von Faceplate1 festgelegt.
- Das Schließen von Faceplate1 schließt auch das Faceplate2.
- Faceplate2 wird bei einem Bildwechsel geschlossen (Folgt aus “IndependentWindow=false“, unabhängig von Parent-Beziehung).

Hinweis Verhalten erst ab TIA Portal V17 Update 4 gültig

Abbildung 3-27

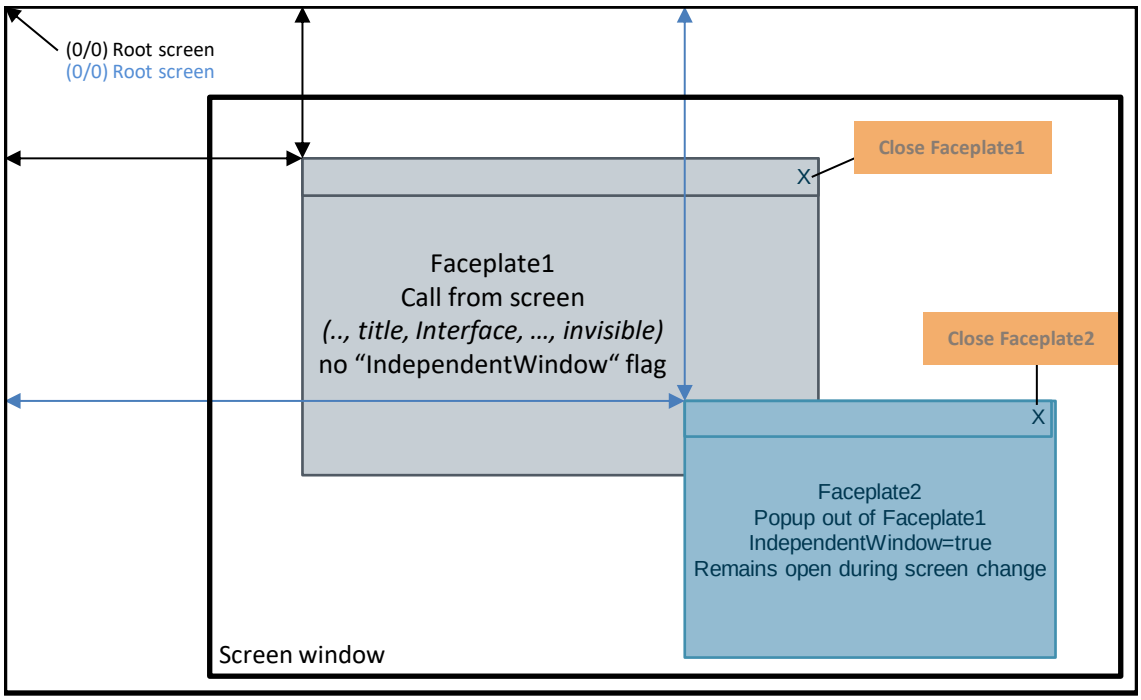


3.7.2 Aufruf in Bildfenster

3.7.2.1 Absolut und persistent (Hauptbild)

- Das aufrufende Faceplate1 wird in einem Bild innerhalb eines Bildfensters aufgerufen.
- Das aufrufende Faceplate1 hat keinen “IndependentWindow“-Parameter.
- Das PopUp-Faceplate2 wird aus Faceplate1 aufgerufen.
- Der “IndependentWindow“-Parameter von Faceplate2 ist “true“.
- Der Koordinatenursprung (0/0) beider Faceplates entspricht dem des (Haupt-) Bilds.
- Faceplate1 und Faceplate2 lassen sich unabhängig voneinander schließen.
- Faceplate2 bleibt bei einem Bildwechsel (vordergründig) geöffnet.

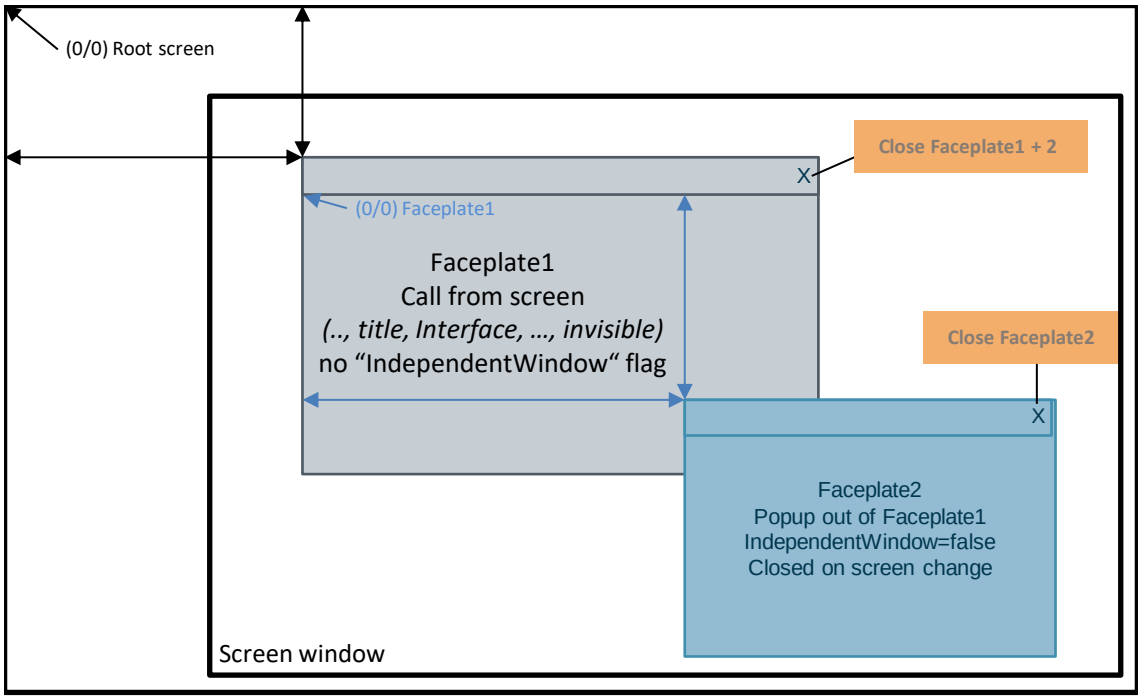
Abbildung 3-28



3.7.2.2 Relativ und temporär (Hauptbild)

- Das aufrufende Faceplate1 wird in einem Bild innerhalb eines Bildfensters aufgerufen.
- Das aufrufende Faceplate1 hat keinen “IndependentWindow“-Parameter.
- Das PopUp-Faceplate2 wird aus Faceplate1 aufgerufen.
- Der “IndependentWindow“-Parameter von Faceplate2 ist “false“.
- Der Koordinatenursprung (0/0) von Faceplate1 entspricht dem des (Haupt) Bilds.
- Der Koordinatenursprung (0/0) von Faceplate2 entspricht dem des Bildinhalts von Faceplate1.
- Das Schließen von Faceplate1 schließt auch das Faceplate2.
- Faceplate2 wird bei einem Bildwechsel geschlossen.

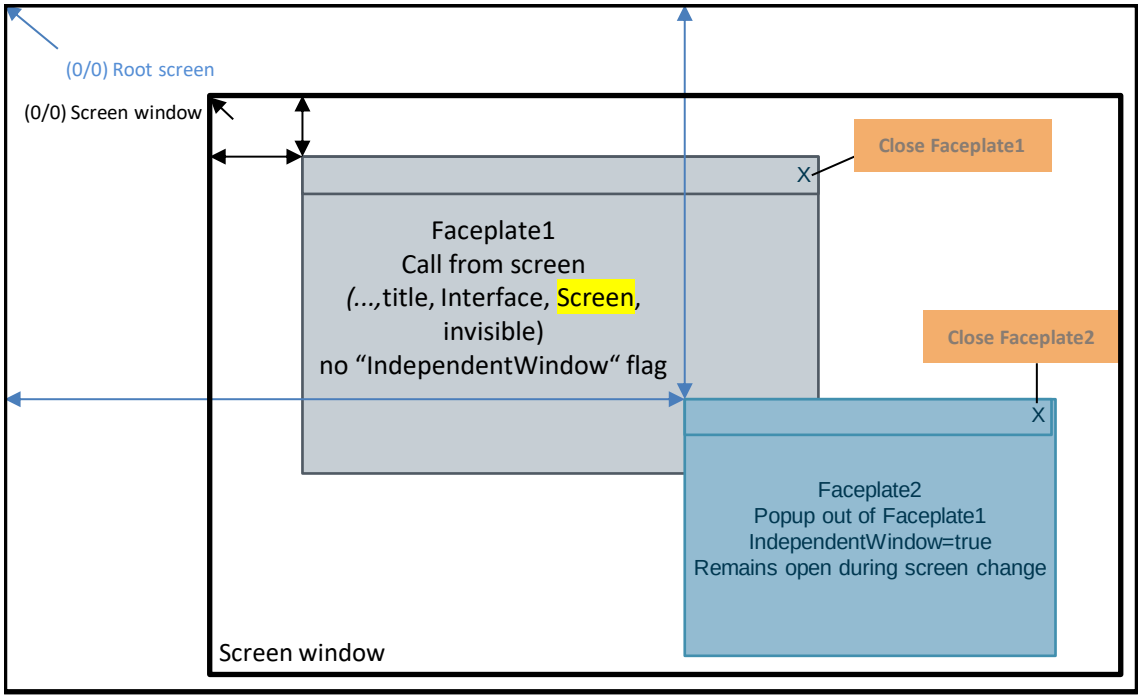
Abbildung 3-29



3.7.2.3 Absolut und persistent (Bildfenster)

- Das aufrufende Faceplate1 wird in einem Bild innerhalb eines Bildfensters aufgerufen.
- Das aufrufende Faceplate1 hat keinen “IndependentWindow“-Parameter.
- In Faceplate1 wird auf dessen Parent-Objekt (Bildfenster) zugegriffen.
  - ⇨ Der Koordinatenursprung (0/0) von Faceplate1 entspricht dem des Bildfensters.
- Das PopUp-Faceplate2 wird aus Faceplate1 aufgerufen.
- Der “IndependentWindow“-Parameter von Faceplate2 ist “true“.
- Der Koordinatenursprung (0/0) von Faceplate2 entspricht dem des (Haupt) Bilds.
- Faceplate1 und Faceplate2 lassen sich unabhängig voneinander schließen.
- Faceplate2 bleibt bei einem Bildwechsel (vordergründig) geöffnet.

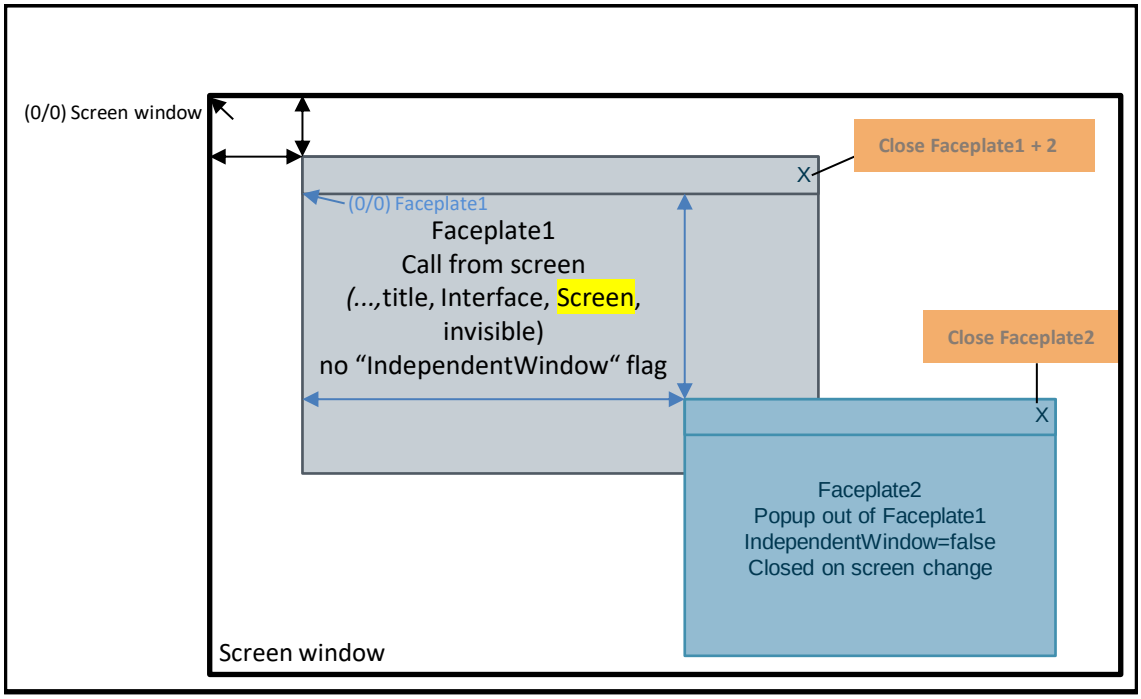
Abbildung 3-30



3.7.2.4 Relativ und temporär (Bildfenster)

- Das aufrufende Faceplate1 wird in einem Bild innerhalb eines Bildfensters aufgerufen.
- Das aufrufende Faceplate1 hat keinen “IndependentWindow“-Parameter.
- In Faceplate1 wird auf dessen Parent-Objekt (Bildfenster) zugegriffen.
  - ⇨ Der Koordinatenursprung (0/0) von Faceplate1 entspricht dem des Bildfensters.
- Das PopUp-Faceplate2 wird aus Faceplate1 aufgerufen.
- Der “IndependentWindow“-Parameter von Faceplate2 ist “false“.
- Der Koordinatenursprung (0/0) von Faceplate2 entspricht dem des Bildinhalts von Faceplate1.
- Das Schließen von Faceplate1 schließt auch das Faceplate2.
- Faceplate2 wird bei einem Bildwechsel geschlossen.

Abbildung 3-31



4 Anhang

4.1 Service und Support

Industry Online Support

Sie haben Fragen oder brauchen Unterstützung?

Über den Industry Online Support greifen Sie rund um die Uhr auf das gesamte Service und Support Know-how sowie auf unsere Dienstleistungen zu.

Der Industry Online Support ist die zentrale Adresse für Informationen zu unseren Produkten, Lösungen und Services.

Produktinformationen, Handbücher, Downloads, FAQs und Anwendungsbeispiele – alle Informationen sind mit wenigen Mausklicks erreichbar:

support.industry.siemens.com

Technical Support

Der Technical Support von Siemens Industry unterstützt Sie schnell und kompetent bei allen technischen Anfragen mit einer Vielzahl maßgeschneiderter Angebote – von der Basisunterstützung bis hin zu individuellen Supportverträgen.

Anfragen an den Technical Support stellen Sie per Web-Formular:
siemens.com/SupportRequest

SITRAIN – Digital Industry Academy

Mit unseren weltweit verfügbaren Trainings für unsere Produkte und Lösungen unterstützen wir Sie praxisnah, mit innovativen Lernmethoden und mit einem kundenspezifisch abgestimmten Konzept.

Mehr zu den angebotenen Trainings und Kursen sowie deren Standorte und Termine erfahren Sie unter:

siemens.de/sitrain

Serviceangebot

Unser Serviceangebot umfasst folgendes:

- Plant Data Services
- Ersatzteilservices
- Reparaturservices
- Vor-Ort und Instandhaltungsservices
- Retrofit- und Modernisierungsservices
- Serviceprogramme und Verträge

Ausführliche Informationen zu unserem Serviceangebot finden Sie im Servicekatalog:

support.industry.siemens.com/cs/sc

Industry Online Support App

Mit der App "Siemens Industry Online Support" erhalten Sie auch unterwegs die optimale Unterstützung. Die App ist für iOS und Android verfügbar:

support.industry.siemens.com/cs/ww/de/sc/2067

4.2 Industry Mall

Die Siemens Industry Mall ist die Plattform, auf der das gesamte Produktportfolio von Siemens Industry zugänglich ist. Von der Auswahl der Produkte über die Bestellung und die Lieferverfolgung ermöglicht die Industry Mall die komplette Einkaufsabwicklung – direkt und unabhängig von Zeit und Ort:
mall.industry.siemens.com

4.3 Links und Literatur

Tabelle 4-1

Nr.	Thema
\1\	Siemens Industry Online Support https://support.industry.siemens.com
\2\	Link auf die Beitragsseite des Anwendungsbeispiels https://support.industry.siemens.com/cs/ww/de/view/109812366
\3\	SIMATIC HMI WinCC Unified Getting Started https://support.industry.siemens.com/cs/de/de/view/109801175
\4\	Handbuch SIMATIC HMI WinCC Unified Engineering V18 https://support.industry.siemens.com/cs/de/de/view/109813308
\5\	SIMATIC WinCC Unified - Tipps und Tricks zur Skripterstellung (JavaScript) https://support.industry.siemens.com/cs/de/de/view/109758536
\6\	SITRAIN-Systemkurs: WinCC Unified & Unified Comfort Panels https://support.industry.siemens.com/cs/ww/de/view/109773211
\7\	SITRAIN-Aufbaukurs: SIMATIC WinCC Unified für PC-Systeme https://support.industry.siemens.com/cs/ww/de/view/109781323
\8\	Multiuser Engineering mit TIA Portal Projekt-Server https://support.industry.siemens.com/cs/de/de/view/109740141
\9\	Das TIA Portal Tutorial Center (Videos) https://support.industry.siemens.com/cs/de/de/view/106656707

4.4 Änderungsdokumentation

Tabelle 4-2

Version	Datum	Änderung
V1.0	01/2023	Erste Ausgabe