SIEMENS

SIMATIC

Openness: Projekterstellung automatisieren

Systemhandbuch

Sicherheitshinweis 1

Liesmich zu TIA Portal Openness 2

Was ist neu in TIA Portal Openness? 3

Grundlagen 4

Einführung 5

Konfigurationen 6

TIA Portal Openness API 7

Export/Import 8

Wesentliche Änderungen 9

Ausdruck der Online-Hilfe

10/2018

Rechtliche Hinweise

Warnhinweiskonzept

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

⚠ GEFAHR
bedeutet, dass Tod oder schwere Körperverletzung eintreten **wird**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

⚠ WARNUNG
bedeutet, dass Tod oder schwere Körperverletzung eintreten **kann**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

⚠ VORSICHT
bedeutet, dass eine leichte Körperverletzung eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

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

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

Qualifiziertes Personal

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

Bestimmungsgemäßer Gebrauch von Siemens-Produkten

Beachten Sie Folgendes:

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

Marken

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

Haftungsausschluss

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

Siemens AG
Division Digital Factory
Postfach 48 48
90026 NÜRNBERG
DEUTSCHLAND

℗ 10/2018 Änderungen vorbehalten

Inhaltsverzeichnis

1 Sicherheitshinweis....11

2 Liesmich zu TIA Portal Openness....13

2.1 Liesmich....13

2.2 Größere Änderungen in TIA Portal Openness V15.1....16

2.3 Bekanntgabe größerer Änderungen in künftigen Releases....19

2.4 Hinweise zum Schreiben von langfristig stabilem Code....20

3 Was ist neu in TIA Portal Openness?....23

4 Grundlagen....25

4.1 Voraussetzungen für TIA Portal Openness....25

4.2 Installation....27
4.2.1 TIA Openness installieren....27
4.2.2 Benutzer der Benutzergruppe "Siemens TIA Openness" hinzufügen....28
4.2.3 Auf das TIA Portal zugreifen....34

4.3 Aufgaben von Openness....35
4.3.1 Einsatzmöglichkeiten....35
4.3.2 Export/Import....36

4.4 Objektliste....37

4.5 Standard-Bibliotheken....41

4.6 Anmerkungen zur Leistung von TIA Portal Openness....42

5 Einführung....43

6 Konfigurationen....45

7 TIA Portal Openness API....49

7.1 Einleitung....49

7.2 Programmierschritte....50

7.3 TIA Portal Openness-Objektmodell....51

7.4 Bausteine und Typen des TIA Portal Openness-Objektmodells....56

7.5 Hierarchie von Hardware-Objekten des Objektmodells....64

7.6 Informationen über installierte TIA Portal Openness-Versionen....66

7.7 Beispielprogramm....67

7.8 Einsatz der Codebeispiele....72

7.9 Allgemeine Funktionen....74
7.9.1 Unterstützung von IntelliSense durch TIA Portal Openness....74
7.9.2 Verbindung zum TIA Portal aufbauen....74
7.9.3 TIA Portal Openness-Firewall....79

7.9.4 Event-Handler....80
7.9.5 Dialoge mit Systemmeldungen programmgesteuert bestätigen....82
7.9.6 Verbindung zum TIA Portal beenden....83
7.9.7 Diagnoseschnittstellen im TIA Portal....84
7.9.8 Exclusive access....90
7.9.9 Transaktionsbehandlung....92
7.9.10 Ein Objekt DirectoryInfo/FileInfo erstellen....95
7.9.11 Unterstützung der Selbstbeschreibung für Attribute, Navigatoren, Aktionen und Dienste....96

7.10 Funktionen der Projekte und Projektdaten....99
7.10.1 Projekt öffnen....99
7.10.2 Projekt anlegen....103
7.10.3 Auf allgemeine Einstellungen des TIA Portals zugreifen....104
7.10.4 Schreibgeschütztes TIA Portal-Projekt öffnen....108
7.10.5 Auf Sprachen zugreifen....109
7.10.6 Objektstruktur und -Attribute ermitteln....111
7.10.7 Auf Software-Ziel zugreifen....113
7.10.8 Auf mehrsprachige Texte zugreifen und sie enumerieren....114
7.10.9 Projektbezogene Attribute lesen....115
7.10.10 Projektgrafik löschen....118
7.10.11 Projekt übersetzen....118
7.10.12 Projekt speichern....121
7.10.13 Projekt schließen....122

7.11 Funktionen für Verbindungen....124
7.11.1 Konfigurierbare Attribute einer Port-zu-Port-Verbindung....124

7.12 Funktionen auf Bibliotheken....127
7.12.1 Funktionen auf Objekte und Instanzen....127
7.12.2 Auf globale Bibliotheken zugreifen....128
7.12.3 Zugriff auf Sprachen der globalen Bibliothek....130
7.12.4 Bibliotheken öffnen....132
7.12.5 Offene Bibliotheken enumerieren....134
7.12.6 Bibliotheken speichern und schließen....134
7.12.7 Bibliothek archivieren und abrufen....136
7.12.8 Globale Bibliotheken erstellen....138
7.12.9 Auf Ordner in einer Bibliothek zugreifen....139
7.12.10 Auf Typen zugreifen....143
7.12.11 Auf Typ-Versionen zugreifen....145
7.12.12 Auf Instanzen zugreifen....150
7.12.13 Auf Kopiervorlagen zugreifen....152
7.12.14 Masterkopie aus einem Projekt in Bibliothek erzeugen....154
7.12.15 Ein Objekt aus einer Masterkopie erstellen....156
7.12.16 Masterkopien kopieren....158
7.12.17 Ermitteln veralteter Typinstanzen....159
7.12.18 Projekt aktualisieren....162
7.12.19 Bibliothek aktualisieren....164
7.12.20 Bibliotheksinhalte löschen....165

7.13 Funktionen für das Aufrufen von Geräten, Netzwerken und Verbindungen....168
7.13.1 Editor "Geräte & Netze" öffnen....168
7.13.2 PLC-Target und HMI-Target abfragen....169
7.13.3 Auf Attribute eines Adressobjekts zugreifen....170
7.13.4 Aufrufen eines Modulkanals....173

7.13.5 Arbeiten mit Assoziationen....175
7.13.6 Mit Zusammensetzungen arbeiten....175
7.13.7 Objektgleichheit prüfen....177
7.13.8 Lese-Operationen für Attribute....178

7.14 Funktionen in Netzwerken....180
7.14.1 Subnetz anlegen....180
7.14.2 Auf Subnetze zugreifen....181
7.14.3 Auf interne Subnetze zugreifen....182
7.14.4 Typkennung von Subnetzen abrufen....183
7.14.5 Attribute eines Subnetzes aufrufen....184
7.14.6 Globales Subnetz löschen....190
7.14.7 Alle Beteiligten eines Subnetzes enumerieren....190
7.14.8 IO-Systeme eines Subnetzes enumerieren....191
7.14.9 Auf Teilnehmer zugreifen....192
7.14.10 Attribute eines Teilnehmers aufrufen....193
7.14.11 Teilnehmer mit einem Subnetz verbinden....197
7.14.12 Teilnehmer von einem Subnetz trennen....197
7.14.13 IO-System erstellen....198
7.14.14 Attribute eines IO-Systems aufrufen....199
7.14.15 IO-Connector mit einem IO-System verbinden....199
7.14.16 Mastersystem oder IO-System einer Schnittstelle abrufen....200
7.14.17 IO-Controller abrufen....201
7.14.18 IO-Connector abrufen....202
7.14.19 IO-Connector von einem IO-System oder einem DP-Mastersystem trennen....202
7.14.20 Attribute eines DP-Mastersystems aufrufen....203
7.14.21 Attribute eines PROFINET IO-Systems aufrufen....204
7.14.22 DP-Mastersystem löschen....205
7.14.23 Profinet IO-System löschen....206
7.14.24 DP-Mastersystem erstellen....206
7.14.25 Port-Verschaltungsinformationen des Port-Geräteelements aufrufen....207
7.14.26 Attribute der Port-Verschaltung....208
7.14.27 Attribute eines Ports aufrufen....211
7.14.28 DP-Mastersysteme eines Subnetzes enumerieren....212
7.14.29 Zugeordnete IO-Connectors enumerieren....213
7.14.30 DP-IO-Connector mit einem DP-Mastersystem verbinden....214

7.15 Funktionen auf Geräten....215
7.15.1 Obligatorische Attribute von Geräten....215
7.15.2 Typkennung von Geräten und Geräteelementen abrufen....216
7.15.3 Erstellen eines Geräts....219
7.15.4 Geräte enumerieren....220
7.15.5 Auf Geräte zugreifen....223
7.15.6 Löschen eines Geräts....225

7.16 Funktionen auf Geräteelementen....227
7.16.1 Obligatorische Attribute von Geräteelementen....227
7.16.2 Ein Geräteelement erstellen und stecken....229
7.16.3 Geräteelemente in einen anderen Steckplatz verschieben....232
7.16.4 Geräteelement kopieren....233
7.16.5 Geräteelement löschen....234
7.16.6 Geräteelemente enumerieren....235
7.16.7 Geräteelemente aufrufen....236
7.16.8 Auf das Geräteelement als Schnittstelle zugreifen....240

7.16.9 Auf Attribute einer I/O-Geräteschnittstelle zugreifen....241
7.16.10 Auf Attribute des IoController zugreifen....243
7.16.11 Auf Attribute des IoConnector zugreifen....244
7.16.12 Auf einen Adresscontroller zugreifen....246
7.16.13 Auf Adressen zugreifen....247
7.16.14 Auf "Hardwarekennung" zugreifen....249
7.16.15 Auf Hardwarekennungscontroller zugreifen....250
7.16.16 Auf Kanäle von Geräteelementen zugreifen....251

7.17 Funktionen auf Daten eines HMI-Gerätes....253
7.17.1 Bilder....253
7.17.1.1 Benutzerdefinierte Bilderordner erzeugen....253
7.17.1.2 Bild aus einem Ordner löschen....253
7.17.1.3 Bildvorlage aus einem Ordner löschen....254
7.17.1.4 Alle Bilder aus einem Ordner löschen....255
7.17.2 Zyklen....256
7.17.2.1 Zyklus löschen....256
7.17.3 Textlisten....256
7.17.3.1 Textliste löschen....256
7.17.4 Grafiklisten....257
7.17.4.1 Grafikliste löschen....257
7.17.5 Verbindungen....258
7.17.5.1 Verbindung löschen....258
7.17.6 Variablentabelle....258
7.17.6.1 Benutzerdefinierte Ordner für HMI-Variablen erzeugen....258
7.17.6.2 Variablen einer HMI-Variablentabelle enumerieren....259
7.17.6.3 Einzelne Variable aus einer HMI-Variablentabelle löschen....260
7.17.6.4 Variablentabelle aus einem Ordner löschen....260
7.17.7 VB-Skripte....261
7.17.7.1 Benutzerdefinierte Ordner für Skripte erzeugen....261
7.17.7.2 VB-Skript aus einem Ordner löschen....262
7.17.8 Benutzerdefinierten Ordner eines Bediengeräts löschen....262

7.18 Funktionen auf Daten eines PLC-Gerätes....263
7.18.1 Status einer PLC ermitteln....263
7.18.2 Auf Parameter einer Online-Verbindung zugreifen....264
7.18.3 PLC von R/H-System online setzen....268
7.18.4 Auf Softwarebehälter über primären PLC eines R/H-Systems zugreifen....270
7.18.5 PLCs eines R/H-Systems laden....271
7.18.6 Funktionen zum Laden von Daten ins PLC-Gerät....277
7.18.6.1 Hardware- und Softwarekomponenten ins PLC-Gerät laden....277
7.18.6.2 PLC starten und stoppen....287
7.18.6.3 Unterstützung von Callbacks....288
7.18.6.4 PLC durch Passwort schützen....290
7.18.6.5 Handhabung bausteingebundener PLC-Passwörter....291
7.18.7 Hardware, Software und Dateien in PLC-Gerät laden....292
7.18.8 Zugriff auf Fingerabdrücke....298
7.18.9 PLC-Software vergleichen....299
7.18.10 PLC-Hardware vergleichen....302
7.18.11 Online-Verbindung zur PLC aufbauen oder trennen....303
7.18.12 Bausteine....305
7.18.12.1 Gruppe "Programmbausteine" abfragen....305
7.18.12.2 Systemgruppe für Systembausteine abfragen....305

7.18.12.3 Systemuntergruppen enumerieren....306
7.18.12.4 Benutzerdefinierte Bausteingruppen enumerieren....308
7.18.12.5 Alle Bausteine enumerieren....309
7.18.12.6 Informationen eines Bausteins/Anwenderdatentyps abfragen....310
7.18.12.7 Einen Baustein schützen und Schutz aufheben....311
7.18.12.8 Baustein löschen....314
7.18.12.9 Gruppe für Bausteine erzeugen....314
7.18.12.10 Gruppe für Bausteine löschen....315
7.18.12.11 Auf Attribute sämtlicher Bausteine zugreifen....316
7.18.12.12 Einen ProDiag-FB anlegen....316
7.18.12.13 Auf Überwachungen und Eigenschaften des ProDiag-FB zugreifen....317
7.18.12.14 ProDiag-FB-Bausteine und -Attribute lesen....319
7.18.12.15 Externe Datei hinzufügen....319
7.18.12.16 Quelle aus Baustein generieren....320
7.18.12.17 Bausteine aus Quelle erzeugen....322
7.18.12.18 Anwenderdatentyp löschen....323
7.18.12.19 Externe Datei löschen....324
7.18.12.20 Bausteineditor starten....325
7.18.13 Technologieobjekte....325
7.18.13.1 Übersicht über Funktionen für Technologieobjekte....325
7.18.13.2 Übersicht über Technologieobjekte und Versionen....326
7.18.13.3 Überblick der Datentypen....328
7.18.13.4 Zusammensetzung von Technologieobjekten abfragen....329
7.18.13.5 Technologieobjekt erstellen....329
7.18.13.6 Technologieobjekt löschen....330
7.18.13.7 Technologieobjekt übersetzen....331
7.18.13.8 Technologieobjekte enumerieren....332
7.18.13.9 Technologieobjekt finden....333
7.18.13.10 Parameter eines Technologieobjekts enumerieren....334
7.18.13.11 Parameter eines Technologieobjekts finden....334
7.18.13.12 Parameter eines Technologieobjekts lesen....335
7.18.13.13 Parameter eines Technologieobjekts schreiben....336
7.18.13.14 S7-1200 Motion Control....337
7.18.13.15 S7-1500 Motion Control....345
7.18.13.16 PID-Regelung....364
7.18.13.17 Zählen....365
7.18.13.18 Easy Motion Control....365
7.18.14 Variablen und Variablentabellen....366
7.18.14.1 Starten des PLC-Variableneditors....366
7.18.14.2 Systemgruppen für PLC-Variablen abfragen....367
7.18.14.3 PLC-Variablentabelle anlegen....367
7.18.14.4 Benutzerdefinierte Gruppen für PLC-Variablen enumerieren....368
7.18.14.5 Benutzerdefinierte Gruppen für PLC-Variablen erzeugen....369
7.18.14.6 Benutzerdefinierte Gruppen für PLC-Variablen löschen....370
7.18.14.7 PLC-Variablentabellen in einem Ordner enumerieren....370
7.18.14.8 Informationen einer PLC-Variablentabelle abfragen....371
7.18.14.9 Zeitpunkt der letzten Änderung einer PLC-Variablentabelle lesen....373
7.18.14.10 PLC-Variablentabelle aus einer Gruppe löschen....373
7.18.14.11 PLC-Variablen enumerieren....374
7.18.14.12 Auf PLC-Variablen zugreifen....375
7.18.14.13 Auf PLC-Konstanten zugreifen....376

7.19 Funktionen auf OPC....379

7.19.1 Konfiguration des sicheren Kommunikationsprotokolls für den OPC UA-Server....379
7.19.2 Sicherheitsrichtlinie für OPC UA festlegen....381

7.20 SiVArc Openness....383
7.20.1 Einleitung....383

7.21 Openness für CP 1604/CP 1616/CP 1626....384

7.22 Openness für SIMATIC Ident....388
7.22.1 Openness für SIMATIC Ident....388
7.22.2 ASM 456....389
7.22.3 ASM 475....395
7.22.4 RF120C....397
7.22.5 RF170C....407
7.22.6 RF180C....413
7.22.7 RF18xC....416
7.22.8 RF615R/RF680R/RF685R....419
7.22.9 MV400/MV500....419

7.23 Ausnahmen....420
7.23.1 Umgang mit Exceptions....420

8 Export/Import....423

8.1 Überblick....423
8.1.1 Grundlagen zum Import/Export....423
8.1.2 Einsatzgebiet von Import/Export....425
8.1.3 Versionsspezifischer SimaticML-Import....426
8.1.4 Bearbeiten der XML-Datei....427
8.1.5 Export von Projektierungsdaten....427
8.1.6 Import von Projektierungsdaten....429

8.2 Import/Export von Projektdaten....431
8.2.1 Grafiksammlung....431
8.2.1.1 Export/Import von Grafiken....431
8.2.1.2 Grafiken eines Projektes exportieren....432
8.2.1.3 Grafiken in ein Projekt importieren....433
8.2.2 Projekttexte....434
8.2.2.1 Export von Projekttexten....434
8.2.2.2 Import von Projekttexten....435

8.3 Import/Export von Daten eines HMI-Geräts....437
8.3.1 Aufbau einer XML-Datei....437
8.3.2 Struktur der Daten für Import/Export....439
8.3.3 Zyklen....443
8.3.3.1 Zyklen exportieren....443
8.3.3.2 Zyklen importieren....444
8.3.4 Variablentabellen....445
8.3.4.1 HMI-Variablentabellen exportieren....445
8.3.4.2 HMI-Variablentabelle importieren....448
8.3.4.3 Einzelne Variable einer HMI-Variablentabelle exportieren....449
8.3.4.4 Einzelne Variable in eine HMI-Variablentabelle importieren....450
8.3.4.5 Besonderheiten beim Export/Import von HMI-Variablen....450
8.3.5 VB-Skripte....452
8.3.5.1 VB-Skripte exportieren....452
8.3.5.2 VB-Skripte aus einem Ordner exportieren....453

8.3.5.3 VB-Skripte importieren....454
8.3.6 Textlisten....455
8.3.6.1 Textlisten aus einem HMI-Gerät exportieren....455
8.3.6.2 Textliste in ein HMI-Gerät importieren....456
8.3.6.3 Erweiterte XML-Formate für Export/Import von Textlisten....457
8.3.7 Grafiklisten....459
8.3.7.1 Grafiklisten exportieren....459
8.3.7.2 Grafikliste importieren....459
8.3.8 Verbindungen....460
8.3.8.1 Verbindungen exportieren....460
8.3.8.2 Verbindungen importieren....461
8.3.9 Bilder....462
8.3.9.1 Übersicht der exportierbaren Bild-Objekte....462
8.3.9.2 Alle Bilder eines HMI-Geräts exportieren....466
8.3.9.3 Bild aus einem Bildordner exportieren....467
8.3.9.4 Bilder in ein HMI-Gerät importieren....469
8.3.9.5 Permanentfenster exportieren....472
8.3.9.6 Permanentfenster importieren....472
8.3.9.7 Alle Bildvorlagen eines HMI-Geräts exportieren....473
8.3.9.8 Bildvorlagen aus einem Ordner exportieren....474
8.3.9.9 Bildvorlagen importieren....476
8.3.9.10 Exportieren eines Pop-up-Bilds....478
8.3.9.11 Importieren eines Pop-up-Bildes....479
8.3.9.12 Exportieren eines Slide-in-Bilds....480
8.3.9.13 Importieren eines Slide-in-Bilds....482
8.3.9.14 Bild mit Eingabemaske exportieren....483
8.3.9.15 Bild mit Eingabemaske importieren....485
8.4 Import/Export von Daten eines PLC-Geräts....489
8.4.1 Bausteine....489
8.4.1.1 XML-Struktur des Abschnitts Bausteinschnittstelle....489
8.4.1.2 Änderungen des Objektmodells und Dateiformat XML....499
8.4.1.3 Bausteine exportieren....501
8.4.1.4 DBs mit Schnappschüssen exportieren....507
8.4.1.5 Bausteine mit Know-how-Schutz exportieren....509
8.4.1.6 Export/Import von SCL-Bausteinen....509
8.4.1.7 Export/Import strukturierter Typen von SCL-Bausteinen....524
8.4.1.8 Export/Import von SCL-Aufrufbausteinen....530
8.4.1.9 F-Bausteine exportieren....547
8.4.1.10 System-Bausteine exportieren....547
8.4.1.11 GRAPH-Bausteine mit mehrsprachigem Text exportieren....548
8.4.1.12 Baustein importieren....549
8.4.1.13 Bausteine/UDT mit offenem Verweis importieren....551
8.4.1.14 Bausteine/UDT für Strukturänderungsobjekte importieren....552
8.4.2 Variablentabellen....554
8.4.2.1 PLC-Variablentabellen exportieren....554
8.4.2.2 PLC-Variablentabelle importieren....555
8.4.2.3 Einzelne Variable oder Konstante aus einer PLC-Variablentabelle exportieren....556
8.4.2.4 Einzelne Variable oder Konstante in eine PLC-Variablentabelle importieren....557
8.4.3 Anwenderdatentyp exportieren....558
8.4.4 Anwenderdatentyp importieren....559
8.4.5 Export von Daten im Format OPC UA XML....560

8.5 Hardware-Daten importieren/exportieren....562
8.5.1 AML-Dateiformat....562
8.5.2 Pruned AML....562
8.5.3 Übersicht über Objekte und Parameter von CAx-Import/-Export....564
8.5.4 Struktur der CAx-Daten zum Import/Export....566
8.5.5 AML-Typkennungen....571
8.5.6 Export von CAx-Daten....574
8.5.7 Export/Import von Submodulen....578
8.5.8 Import von CAx-Daten....582
8.5.9 Ausnahmen beim Import und Export von CAx-Daten....584
8.5.10 Round-Trip-Geräte und -Module....585
8.5.11 Export/Import-Topologie....588
8.5.12 Export eines Geräteelements....590
8.5.13 Import eines Geräteobjekts....592
8.5.14 Export/Import eines Geräts mit festgelegter Adresse....595
8.5.15 Export/Import eines Geräts mit Kanälen....598
8.5.16 Export von Geräteelementobjekten....600
8.5.17 Import von Geräteelementobjekten....604
8.5.18 Export/Import von GSD/GSDML-basierten Geräten und Geräteelementen....607
8.5.19 Exportieren/Importieren von Subnetzen....612
8.5.20 Exportieren/Importieren von PLC-Variablen....619
8.5.21 Export/Import von IO-Systemen....621
8.5.22 Exportieren/Importieren mehrsprachiger Kommentare....623
8.5.23 AML-Attribute im Vergleich zu TIA Portal Openness-Attributen....625

9 Wesentliche Änderungen....629

9.1 Größere Änderungen in TIA Portal Openness V15....629

9.2 Die wichtigsten Änderungen in V14 SP1....632
9.2.1 Die wichtigsten Änderungen in V14 SP1....632
9.2.2 Wesentliche Änderungen im Objektmodell....635
9.2.3 Änderungen an der Pilotfunktionalität....639
9.2.4 Änderungen bei Export und Import....644
9.2.4.1 Änderungen bei Export und Import....644
9.2.4.2 Änderungen in der API....644
9.2.4.3 Schemaerweiterung....645
9.2.4.4 Schemaänderungen....648
9.2.4.5 Verhaltensänderungen....651
9.2.4.6 Änderungen an Bausteinattributen....662

9.3 Die wichtigsten Änderungen in V14....664
9.3.1 Wesentliche Änderungen des Objektmodells....664
9.3.2 Vor dem Hochrüsten einer Anwendung auf TIA Portal Openness V14....666
9.3.3 Wesentliche Stringänderungen....667
9.3.4 Importieren von Dateien, die mit TIA Portal Openness V13 SP1 und älter erzeugt wurden....670

Index....673

Sicherheitshinweis

1

Sicherheitsinformationen

Siemens bietet Produkte und Lösungen mit Industrial Security-Funktionen zur Unterstützung des sicheren Betriebs von Anlagen, Systemen, Maschinen, Geräten und/oder Netzwerken.

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

Der Kunde ist dafür verantwortlich, den unberechtigten Zugriff auf seine Anlagen, Systeme, Maschinen und Netzwerke zu verhindern. Systeme, Maschinen und Komponenten dürfen nur in dem erforderlichen Umfang und mit entsprechenden Sicherheitsmaßnahmen (z. B. Firewalls und Netzwerksegmentierung) mit dem Unternehmensnetzwerk oder dem Internet verbunden werden.

Außerdem sollten die Richtlinien von Siemens zu entsprechenden Sicherheitsmaßnahmen berücksichtigt werden. Weitere Informationen zu Industrial Security finden Sie unter

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

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

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

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

Liesmich zu TIA Portal Openness 2

2.1 Liesmich

Sicherheitsmaßnahmen für TIA Portal Openness-Anwendungen

Es ist empfehlenswert,

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

Hardware-Parameter

Eine Beschreibung der Hardware-Parameter steht im Installationsordner von TIA Portal unter Siemens\Automation\Portal V15.1\PublicAPI\V15.1\HW Parameter Description \Openness_hardware_parameter_description.pdf zur Verfügung.

Kopieren einer TIA Portal Openness-Anwendung

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

Abhilfe:

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

Unterstützung bestimmter Funktionen in einem TIA Portal-Projekt

Mehrbenutzerbetrieb

TIA Portal Openness unterstützt keine administrativen Mehrbenutzeroperationen. Der Grund dafür ist, dass der Einsatz von TIA Portal Openness in Mehrbenutzerprojekten nicht empfohlen wird. Beachten Sie, dass bestimmte Aktionen in TIA Portal Openness den von der Benutzeroberfläche des TIA Portals vorgegebenen Mehrbenutzer-Arbeitsfluss sogar stören können. Wenn Sie Änderungen dennoch mit TIA Portal Openness vornehmen möchten, exportieren Sie zuvor das Mehrbenutzerprojekt in ein Einzelbenutzerprojekt.

Fehlersicherheit

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

Verbesserung der Leistung von TIA Portal Openness

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

Threadsicherer Programmcode

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

Exportverhalten von Bildschirmelementen mit aktiviertem Style

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

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

Dieses Problem kann wie folgt behoben werden:

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

Kopieren von S7-1500 Technologieobjekten für die Bewegungssteuerung

Das Kopieren von TO_CamTrack, TO_OutputCam oder TO_MeasuringInput aus der Projektbibliothek oder globalen Bibliothek in das Projekt ist nicht möglich.

Importieren von ASi-Slaves über AML

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

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

Exportieren und Importieren von Funktionstasten

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

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

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

Zugriff auf ein Gerät, das online ist

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

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

Instanzenspezifische Attribute beim Importieren von Bausteinen über TIA Openness

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

Informationen zu bestimmten Funktionen

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

- Projekt archivieren/wiederherstellen
- Beobachtungstabellen exportieren/importieren

2.2 Größere Änderungen in TIA Portal Openness V15.1

Änderungen

Wenn Sie die Hinweise zur versionsübergreifenden Programmierung beachtet haben und Ihr Projekt nicht auf V15.1 aktualisieren, laufen Ihre Anwendungen ohne jede Einschränkung auf jedem Rechner, selbst wenn nur ein TIA Portal V15.1 installiert ist.

Wenn Sie Ihr Projekt auf V15.1 aktualisieren, ist es notwendig, Ihre Anwendung mit der SiemensEngineering.dll von V15.1 neu zu übersetzen. In manchen Fällen kann es erforderlich sind, den Code Ihrer Anwendung anzupassen.

Typkennungen

Die Typkennung für Baugruppenträger und Geräte der Art "PC mit Ports" sowie "Ethernet-Gerät mit Ports" wurde geändert.

PC mit Ports	Typkennung vor V15.1	Typkennung ab V15.1
Gerät (device)	System:DesktopPC.Device	System:Device.DesktopPC
Baugruppenträger (rack)	System:DesktopPC.Rack	System:Rack.DesktopPC
Geräteelement	System:DesktopPC.Port<X> <X> denotes the number of ports	System:DeviceItem.EthernetDe-vice.Port<X> <X> denotes the number of ports

Ethernet-Geräte mit Ports		
Gerät (device)	System:DummyPC.Device	System:Device.EthernetDevice
Baugruppenträger (rack)	System:Rack.DummyPC	System:Rack.EthernetDevice
Geräteelement	System:DummyPC.Port<X> <X> denotes the number of ports	

Ausfälle beim Versuch, mit dem TIA Portal zu verbinden

Führt der Versuch, eine Verbindung zum TIA Portal herzustellen, zu einem Ausfall, ist die daraufhin angezeigte Meldung nun spezifischer.

Thread-übergreifender Betrieb

Der Zugriff auf Openness-Objekte ist nicht inhärent threadsicher.
Wenn Sie Multithreading zur Verbesserung der Performance Ihrer Openness-Anwendung einsetzen, wird die Erstellung der Instanz des TIA Portals mit MTA empfohlen.
Wenn TIA Portal innerhalb eines STA-Thread erstellt oder angehängt wird, sollte auf alle Openness-Objekte in Zusammenhang mit dieser Instanz des TIA Portals aus demselben STA-Thread zugegriffen werden; andernfalls wird eine Ausnahme ausgelöst.

Submodule haben nicht die Attribute Author und TypeName.

Die Attribute Author und TypeName wurden von Submodulen, die nicht gesteckt werden können, entfernt.

Öffnen einer globalen Bibliothek

Ab TIA Portal Openness V15.1 kann eine über Openness, unabhängig von einem bestehenden Vorschaumodus der Bibliothek, eine globale Bibliothek geöffnet werden.

Code bei Anwendungsabbruch

Im Fall eines Codes bei Anwendungsabbruch geschieht Folgendes:

- Bis zu TIA Portal Openness V15 erhalten Sie einen Ausnahmefehler ohne Wiederherstellbarkeit
- Ab TIA Portal Openness V15.1 erhalten Sie eine EngineeringRuntimeException oder eine EngineeringTargetInvocationException, wenn der Fehlercode bekannt ist, oder einen Ausnahmefehler ohne Wiederherstellbarkeit, wenn der Fehlercode unbekannt ist.

Schemaerweiterung für namenlose Parameter

SCL-Bausteine können importiert werden, selbst wenn ENO bei einem nicht formellen Aufruf genutzt wird.

Header indizierter Parameter

Ab TIA Portal Openness V15.1 kann der Header indizierter Parameter nicht über Openness geändert werden.

Attribut TransmissionRateAndDuplex

Einige falsche Enum-Werte für das Attribut TransmissionRateAndDuplex wurden korrigiert.

Attribut AutoNumber für Bausteine mit Know-how-Schutz

Ab V15.1 kann das Attribut AutoNumber nicht über TIA Portal Openness geändert werden, wenn ein Baustein über Know-how-Schutz verfügt.

Anzahl der von Schnittstelle ChannelInfo gelisteten Kanäle

Bis TIA Portal Openness V15 zeigt die Schnittstelle ChannelInfo die Anzahl der verfügbaren Kanälen bei manchen Modulen nicht richtig an.

Zugriff auf ProDiag-FB-Attribute

Auf die folgenden Attribute eines ProDiag-Funktionsbausteins kann über TIA Portal Openness zugegriffen werden:

- Version
- Erstwerterfassung
- Nutzung zentraler Zeitstempel

Import/Export fehlersicherer Bausteine

Der Import fehlersicherer Bausteine aus früheren Versionen ist nicht möglich.

Der Export von vom System generierten fehlersicheren Bausteinen wird ab TIA Portal Openness V15.1 verhindert.

R/H-Systeme

Der Download für R/H-Geräte ist auf dem Gerät verfügbar.

Online-Dienst ist für R/H-Geräte nicht verfügbar.

Für PLC2 eines R/H-Systems ist SoftwareContainer nicht verfügbar.

2.3 Bekanntgabe größerer Änderungen in künftigen Releases

Bekanntgabe von Änderungen

Die TIA Portal Openness API wird in einer späteren Version geändert. Es besteht keine Notwendigkeit, den Code Ihrer Anwendung sofort zu ändern, weil Anwendungen auf der Grundlage früherer Releases ohne jede Einschränkung laufen. Aber für neue Anwendungen empfiehlt es sich, die neue Funktionalität zu nutzen und die Neucodierung Ihrer Anwendung zu planen, da ab V17 die folgenden Methoden nicht mehr unterstützt werden.

- Typ von Zusammensetzungen

Typ von Zusammensetzungen

Die folgenden Typen ändern sich, um das Momentabbildverhalten anzuzeigen:

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

Simocode Openness

Bitte beachten Sie, dass zwar alle Parameter der Wahrheitstabelle in Openness V15.1 genutzt werden können, sich ihre Implementierung in V16 jedoch ändern wird.

Anstatt jedes Bit einzeln festzulegen, wird es möglich sein, alle Output-Bits einer Wahrheitstabelle mit einer schnellen Array-Operation einzustellen.

Wenn Sie Ihre bestehenden Openness-Skripte in V16 weiterverwenden möchten, müssen Sie Ihren Code ggf. an die neuen Gegebenheiten anpassen.

2.4 Hinweise zum Schreiben von langfristig stabilem Code

Versionswechsel

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

Registry-Pfad und Datei appconfig

Modifikationen sind notwendig, um Registry-Pfad und Datei appconfig zu ändern. Beispiel:
“C:\Programme\Siemens\Automation\Portal V14\PublicAPI\V14 SP1\Siemens.Engineering.dll”
muss geändert werden in
“C:\Programme\Siemens\Automation\Portal V15\PublicAPI\V14 SP1\Siemens.Engineering.dll”.

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

Installationspfad

Modifikationen sind notwendig, um den Installationspfad des TIA Portals zu ändern. Beispiel:
“C:\Programme\Siemens\Automation\Portal V14\PublicAPI\V14 SP1\Siemens.Engineering.dll”
muss geändert werden in
“C:\Programme\Siemens\Automation\Portal V15\PublicAPI\V14 SP1\Siemens.Engineering.dll”.

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

Pfad von AmiHost

Modifikationen sind notwendig, um den Pfad von AmiHost zu ändern. Beispiel:
“C:\Programme\Siemens\Automation\Portal V14\bin\Siemens.Automation.Cax.AmiHost.exe”
muss geändert werden in “C:\Programme\Siemens\Automation\Portal V15\bin \Siemens.Automation.Cax.AmiHost.exe”.

Um langfristig stabilen Code zu schreiben, muss der Pfad von AmiHost konfigurierbar sein.

Erweiterungen von Projektdateien und Bibliotheken des TIA Portals

Modifikationen sind notwendig, um die Erweiterungen der Projektdateien und Bibliotheken des TIA Portals zu ändern. Beispiel:
*.ap14
muss geändert werden in
*.ap15.

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

Öffnen eines Projekts

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

Hierarchie beim Vergleichen, Übersetzen oder Herunterladen von Ergebnissen

Die Hierarchie und/oder die Reihenfolge beim Vergleichen, Übersetzen oder Herunterladen von Ergebnissen kann sich versionsübergreifend ändern.

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

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

Was ist neu in TIA Portal Openness? 3

Neues TIA Portal Openness-Objektmodell

Die folgenden neuen Funktionen und Innovationen sind in TIA Portal Openness V15.1 verfügbar. Weitere Einzelheiten zu den verschiedenen Themen finden Sie in den einzelnen Abschnitten der Produktdokumentation.

- Openness-DLLs von V14 SP1, V15 und V15.1 im Lieferumfang
  Da die Openness-DLLs von V14 SP1, V15 und V15.1 im Lieferumfang enthalten sind, laufen Anwendungen, die auf V14 SP1 und V15 basieren, auch in V15.1 ohne Änderung. Um die Funktionen von V15.1 zu nutzen, müssen Sie die DLL von V15.1 einbinden und die Anwendung neu übersetzen.
- Siemens.Engineering.dll-Dateien stehen für V14 SP1, V15 und V15.1 zur Verfügung. Sie befinden sich im Installationsverzeichnis unter "PublicAPI\[version]\".
  Beispielsweise ist die dll-Datei für V14 SP1 unter "C:\Program Files\Siemens\Automation \Portal V15_1\PublicAPI\V14 SP1\Siemens.Engineering.dll“ zu finden.
- Zugriff auf Projekte
  Über Openness können in einer Instanz von TIA Portal mehrere Projekte geöffnet werden. Projekte können über Openness archiviert und wiederhergestellt werden.
  Bei UMAC-geschützten Projekten ermöglicht Openness Lesezugriff auf Objekte. Bei dieser Art von Zugriff treffen dieselben Einschränkungen wie bei einem Benutzer mit der Berechtigung "Nur lesen" zu.
- Online/offline-Vergleich
  Über Openness ist der Online/offline-Vergleich von Daten möglich.
- Zugriff auf Schutzstufen
  Eine Schutzstufe kann für Bausteine eingerichtet bzw. entfernt werden.
- Zugriff auf Fingerabdrücke
  Der Fingerabdruck kann für Bausteine und PLC-Datentypen (UDTs) abgefragt werden.
- Zugriff auf Namen
  Der Name kann für Bausteine, Datenbausteine und UDTs festgelegt werden.
- Fehlertoleranter Import
  Zusätzlich zum nach wie vor nutzbaren strengen Import ist nun ein fehlertoleranter Import verfügbar. Der Import ist beispielsweise immer noch möglich, wenn verknüpfte Benutzerdatentypen oder aufgerufene Bausteine nicht übereinstimmen.
- Export und Import
  Beobachtungstabellen sowie Krafttabellen können exportiert und importiert werden. Schnappschüsse der aktuellen Werte können als XML aus einer Offline-DB exportiert werden. Dies bedeutet, dass verschiedene Schnappschüsse anhand von XML-Dateien miteinander verglichen werden können.
- ET200SP
  Lese- und Schreibzugriff ist für die meisten Attribute der ET200SP-Module möglich.

- R/H-Systeme
  Download auf die Primär-PLC und Sicherungs-PLC ist für R/H-Systeme möglich.
- Fehlersichere PLC
  Der Upload ist jetzt auch indirekt über NAT-Router möglich.
  Rezepte, Archive, Benutzerdateien und die Passwörter der Schutzstufen werden beim Upload ebenfalls berücksichtigt.

Weitere Informationen zu Änderungen am Objektmodell finden Sie unter Größere Änderungen in TIA Portal Openness V15.1 (Seite 16).

Siehe auch

TIA Portal Openness-Objektmodell (Seite 51)

Bekanntgabe größerer Änderungen in künftigen Releases (Seite 19)

Größere Änderungen in TIA Portal Openness V15.1 (Seite 16)

Größere Änderungen in TIA Portal Openness V15 (Seite 629)

Die wichtigsten Änderungen in V14 SP1 (Seite 632)

Die wichtigsten Änderungen in V14 (Seite 664)

Grundlagen

4

4.1 Voraussetzungen für TIA Portal Openness

Voraussetzungen für den Einsatz von TIA Openness-Anwendungen

- Auf dem PC ist ein auf dem TIA Portal basierendes Produkt installiert, zum Beispiel "STEP 7 Professional" oder "WinCC Professional".
- TIA Openness ist auf dem PC installiert.
  Siehe TIA Openness installieren (Seite 27)

Unterstützte Windows-Betriebssysteme

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

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

Voraussetzungen für die Programmierung von TIA Portal Openness-Anwendungen

- Microsoft Visual Studio 2015 Update 1 oder höher mit .Net 4.6.2

Notwendige Kenntnisse des Benutzers

- Kenntnisse als Systemtechniker
- Fortgeschrittene Kenntnisse in Microsoft Visual Studio 2015 Update 1 oder höher mit .Net 4.6.2
- Fortgeschrittene Kenntnisse in C# / VB.net und .Net
- Benutzerkenntnisse des TIA Portals

TIA Portal Openness-Remoting-Kanäle

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

Hinweis

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

Für den IpcChannel sind die folgenden Attribute festgelegt:

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

Siehe auch

Benutzer der Benutzergruppe "Siemens TIA Openness" hinzufügen (Seite 28)

4.2 Installation

4.2.1 TIA Openness installieren

Einführung

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

Voraussetzungen

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

Hinweis

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

Vorgehensweise

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

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

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

Ergebnis

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

Hinweis

Sie haben mit dem Add-on-Paket "TIA Openness" weiterhin keinen Zugriff auf das TIA Portal. Sie müssen Mitglied der Benutzergruppe "Siemens TIA Openness" sein (siehe Benutzer der Benutzergruppe "Siemens TIA Openness" hinzufügen (Seite 28)).

4.2.2 Benutzer der Benutzergruppe "Siemens TIA Openness" hinzufügen

Einleitung

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

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

Vorgehensweise

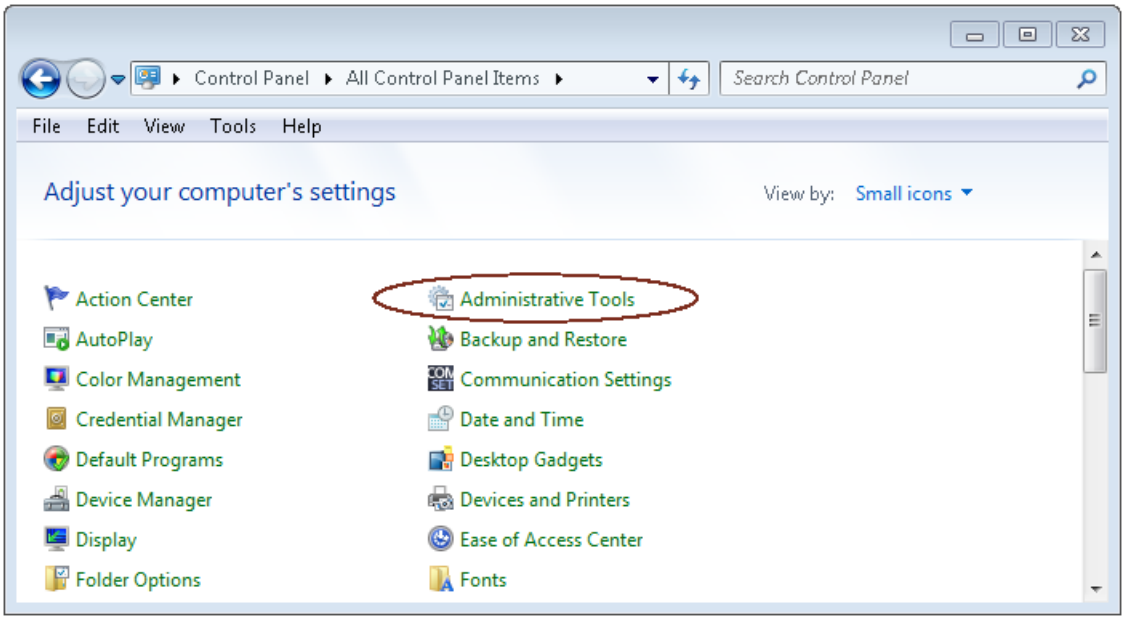
Wenn Sie der Benutzergruppe "Siemens TIA Openness" einen Benutzer hinzufügen möchten, müssen Sie auf Anwendungen aus Ihrem Betriebssystem zurückgreifen. Das TIA Portal unterstützt diesen Vorgang nicht.

Hinweis

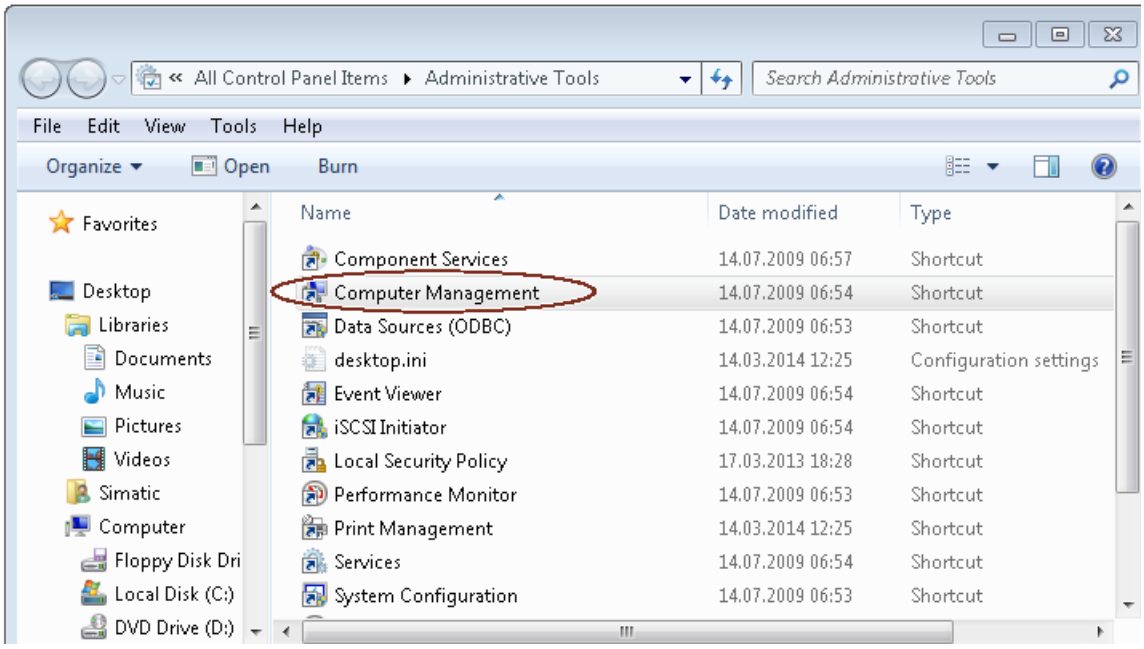
Je nach Konfiguration Ihrer Domain oder Ihres Rechners müssen Sie sich möglicherweise mit Administratorrechten anmelden, um die Benutzergruppe zu erweitern.

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

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

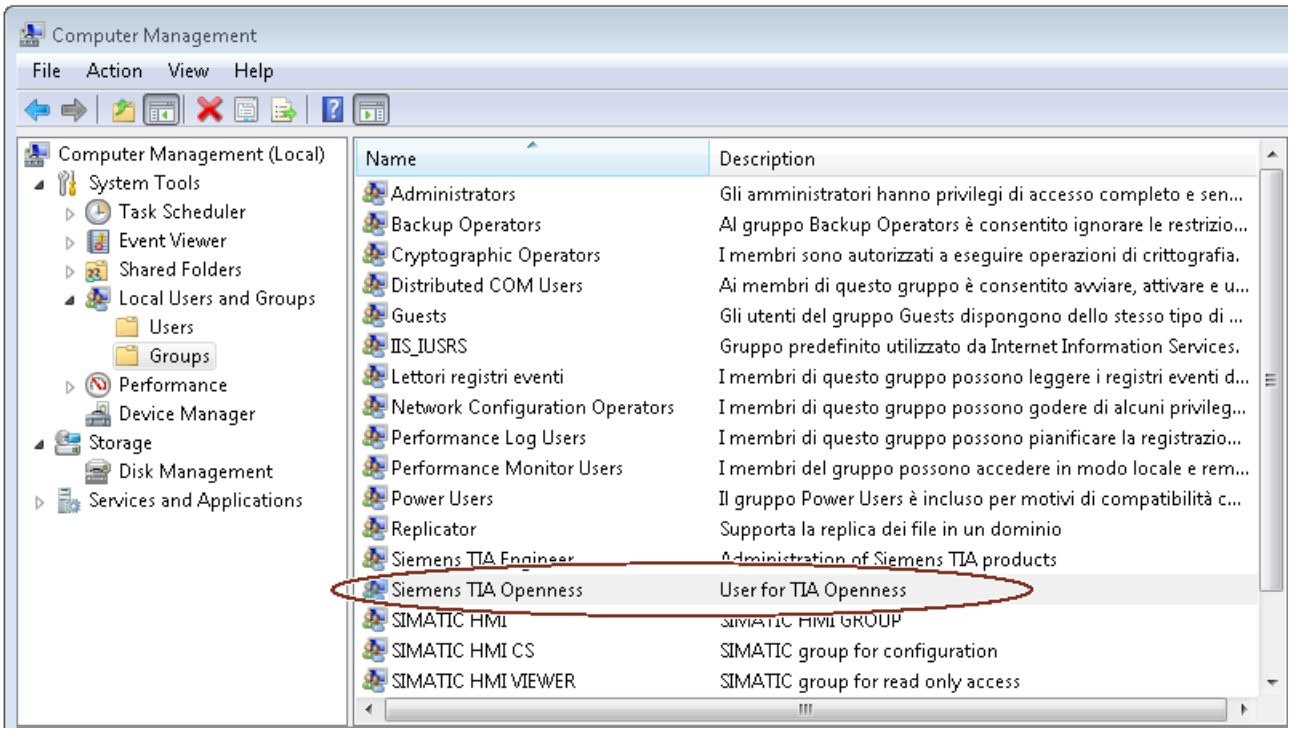


3. Klicken Sie auf "Computer Management", um den gleichnamigen Konfigurationsdialog zu öffnen.

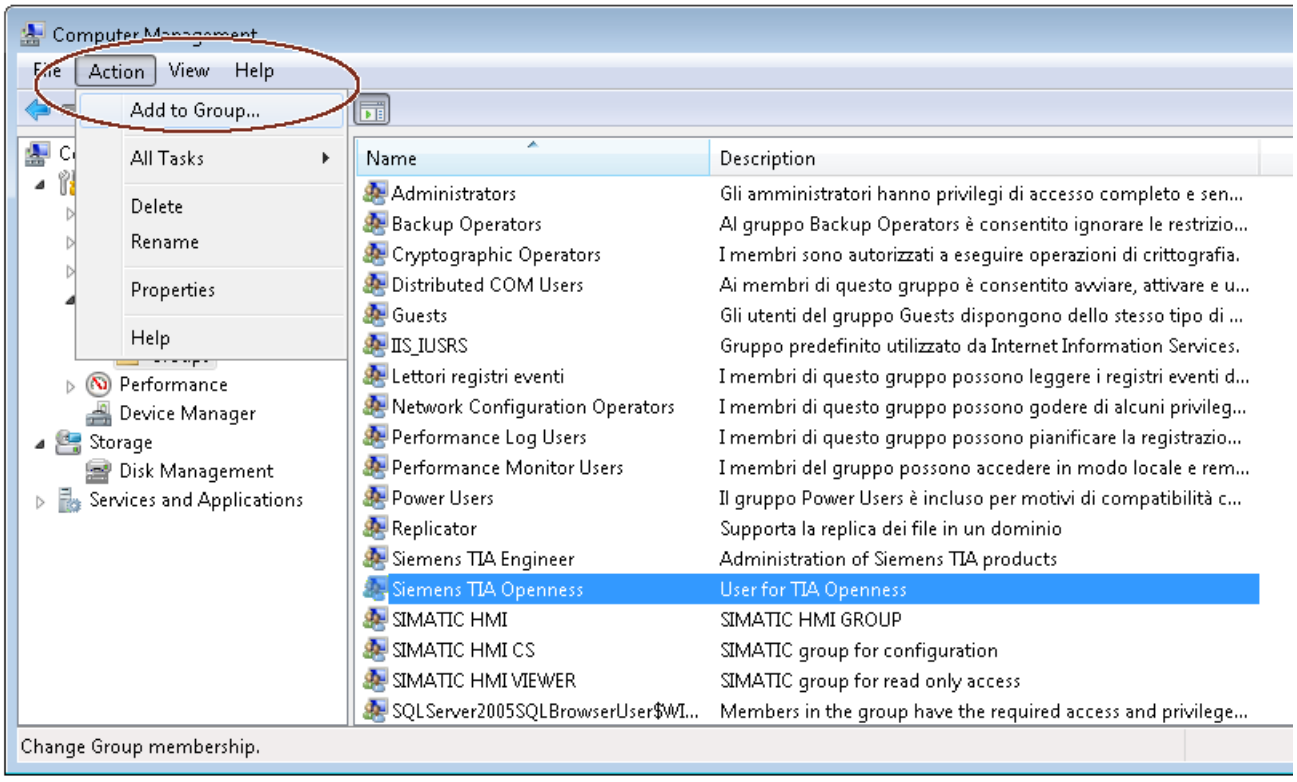


4. Wählen Sie "Local Users and Groups > Groups" aus, um alle erzeugten Benutzergruppen anzuzeigen.

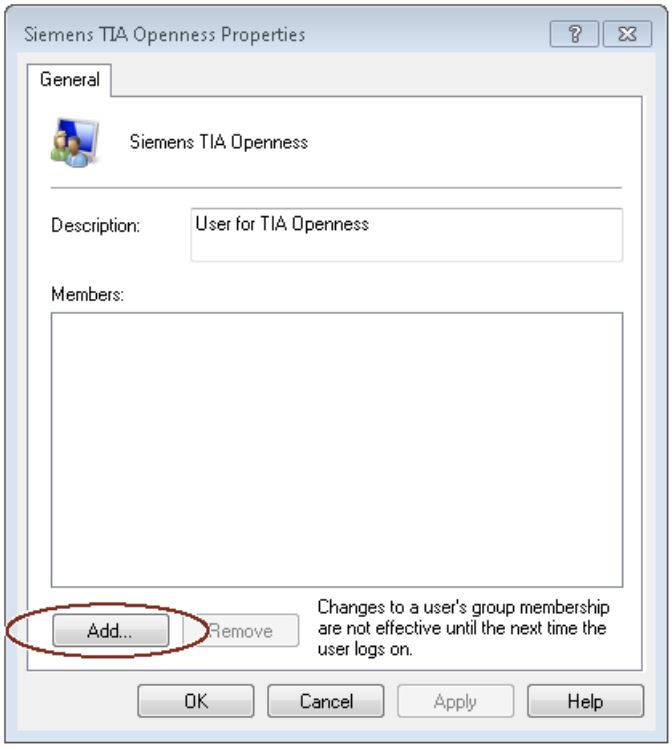
5. Markieren Sie den Eintrag "Siemens TIA Openness" in der Liste der Benutzergruppen im rechten Fenster.



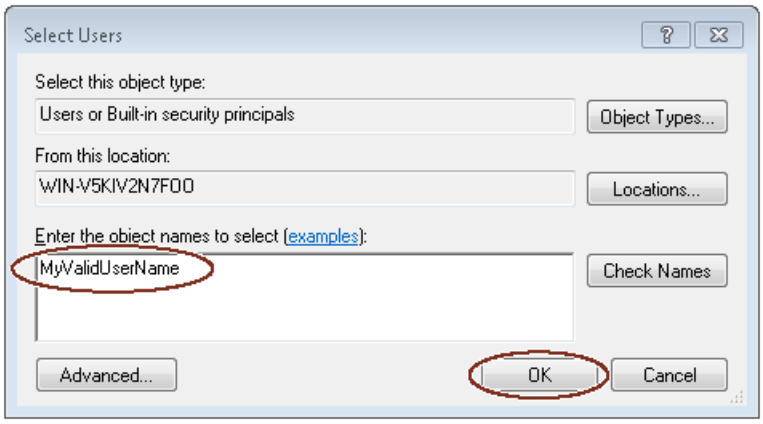
6. Wählen Sie den Menübefehl "Action > Add to Group..." aus.



Der Attributedialog der Benutzergruppe öffnet sich:



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



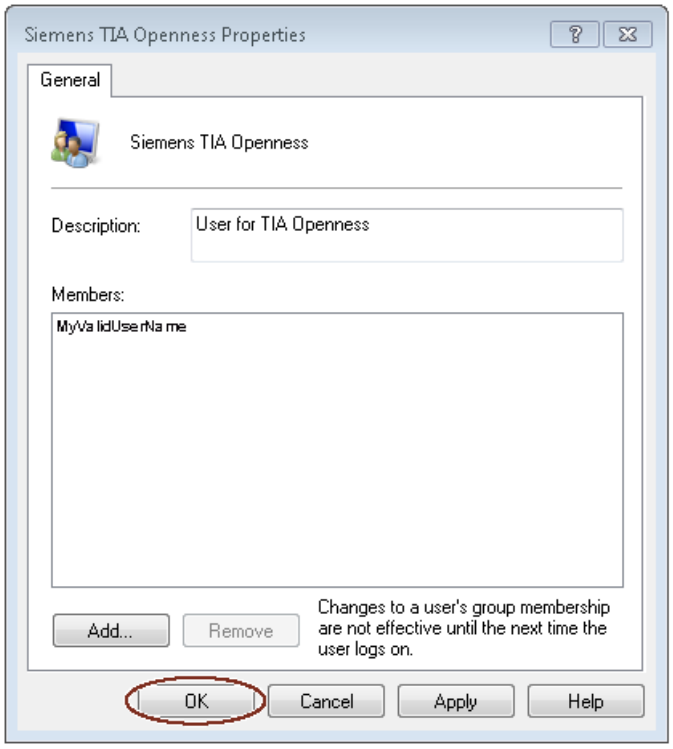
8. Geben Sie einen gültigen Benutzernamen in das Eingabefeld ein.

Hinweis

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

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

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

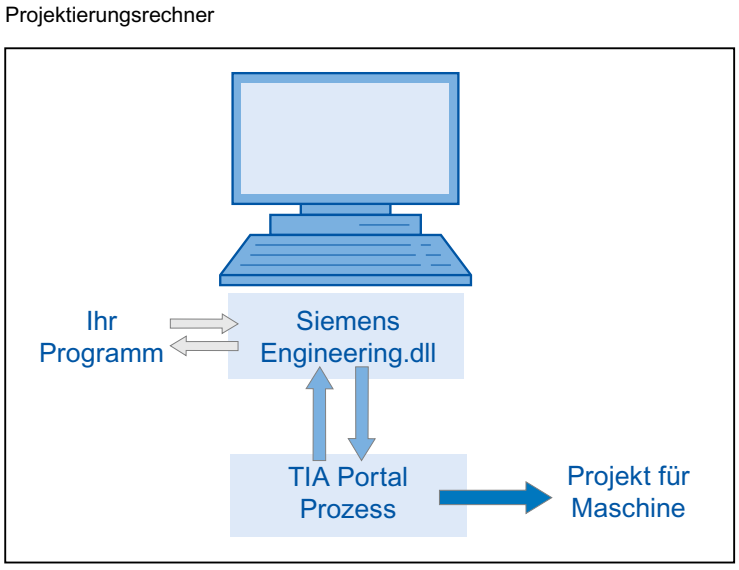


   Zusätzliche Benutzer registrieren Sie durch Klicken auf die Schaltfläche "Add".

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

4.2.3 Auf das TIA Portal zugreifen

Übersicht



Vorgehensweise

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

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Verbindung zum TIA Portal beenden (Seite 83)

4.3 Aufgaben von Openness

4.3.1 Einsatzmöglichkeiten

Einleitung

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

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

- Projektdaten
- PLC-Daten
- HMI-Daten

Hinweis

Sie dürfen die Schnittstelle TIA Portal Openness API nicht dafür nutzen, Prüfungen durchzuführen oder Daten für die Abnahme/Freigabe eines fehlersicheren Systems zu erzeugen. Die Abnahme/Freigabe darf nur mit einem Sicherheitsvordruck unter Verwendung des Add-on-Pakets STEP 7 Safety oder mit dem Funktionstest durchgeführt werden. Die TIA Portal Openness API ist kein Ersatz.

Auf das TIA Portal zugreifen

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

Auf Projekte und Projektdaten zugreifen

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

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

4.3.2 Export/Import

Einleitung

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

Verwendung

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

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

4.4 Objektliste

Einleitung

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

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

Objekte

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

Tabelle 4-1 Bilder

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

Tabelle 4-2 Bildobjekte

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

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

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

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

Tabelle 4-3 Dynamisch

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

Tabelle 4-4 Zusätzliche Objekte

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

Tabelle 4-5 Variablen

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

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

Verbindungen

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

Tabelle 4-6 Listen

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

Tabelle 4-7 Texte

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

4.5 Standard-Bibliotheken

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

```
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 Anmerkungen zur Leistung von TIA Portal Openness

Stammobjekt

In Importdateien können Sie mehrere Stammobjekte angeben.

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

TIA Portal Openness-Funktionen

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

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

Einführung

5

Einleitung

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

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

Mit TIA Portal Openness können Sie folgende Aktionen durchführen:

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

Hinweis

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

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

Hinweis

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

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

Verwendung

Die TIA Portal Openness-Schnittstelle dient folgenden Zwecken:

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

Verwendung von Standardwerten aus dem Bereich der Automatisierungstechnik

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

Bereitstellen von Projektdaten des TIA Portals für externe Anwendungen

- Durch Exportieren von Projektdaten

Sichern von Wettbewerbsvorteilen durch effizientes Engineering

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

Gemeinsam an Projektdaten arbeiten

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

Siehe auch

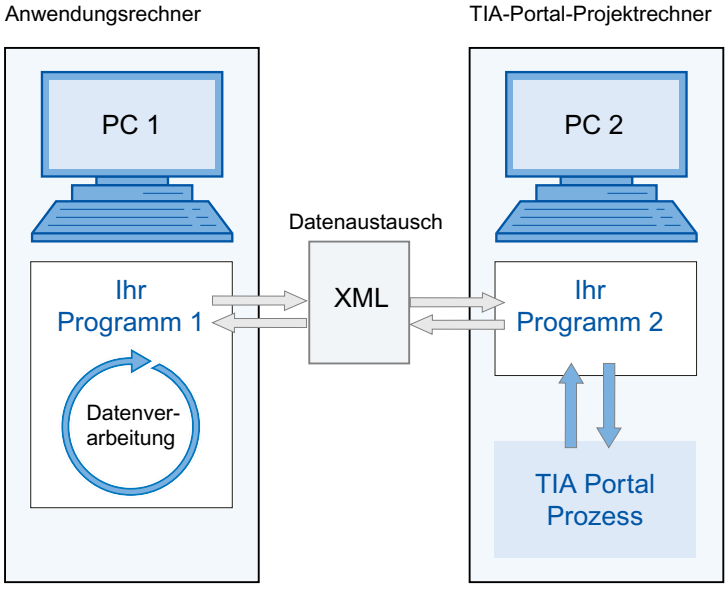
Konfigurationen (Seite 45)

Konfigurationen

6

Sie können mit zwei Varianten von TIA Portal Openness arbeiten:

Die Anwendung und das TIA Portal befinden sich auf unterschiedlichen Computern



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

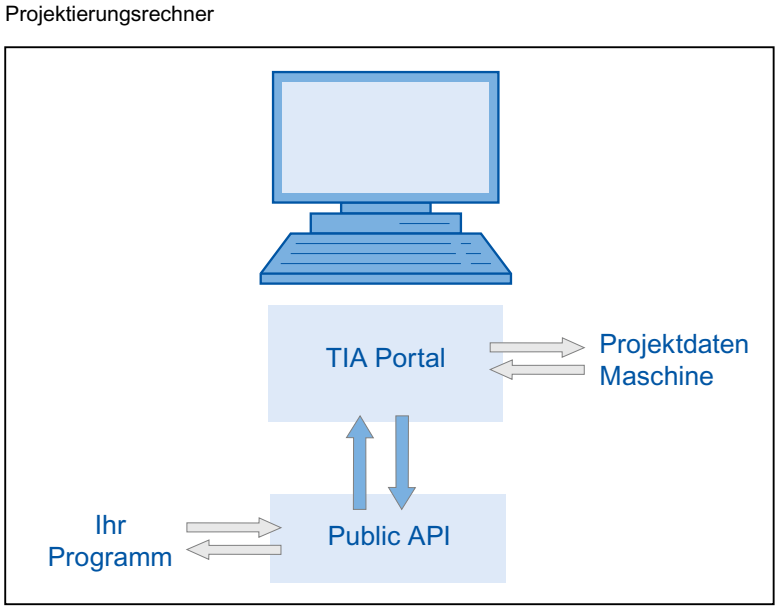
  Hinweis

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

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

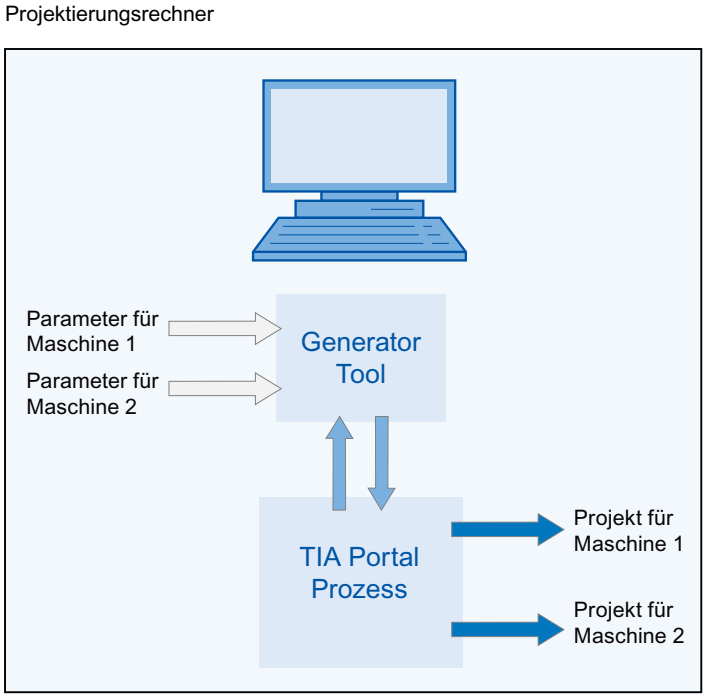
- Sie können ausgetauschte Dateien zu Verifizierungszwecken archivieren.
- Ausgetauschte Daten können an unterschiedlichen Standorten und zu unterschiedlichen Zeiten verarbeitet werden.

Die Anwendung und das TIA Portal befinden sich auf demselben Computer



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

Typischer Anwendungsbereich ist der modulare Maschinenbau



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

TIA Portal Openness API

7

7.1 Einleitung

Übersicht

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

Hinweis

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

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

Beispielprogramm

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

Funktionen

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

Siehe auch

Einsatzmöglichkeiten (Seite 35)

Objektliste (Seite 37)

7.2 Programmierschritte

Übersicht

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

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

Hinweis

Zulässige Strings

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

Siehe auch

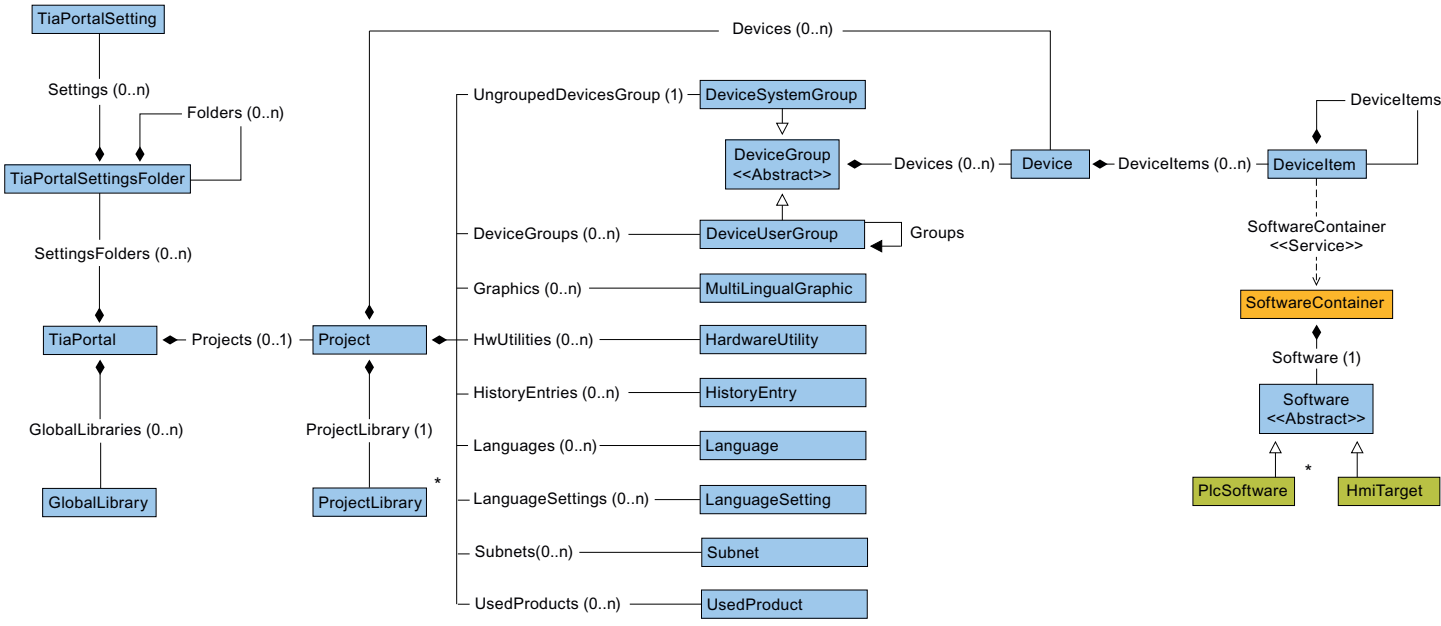
Beispielprogramm (Seite 67)

Einsatz der Codebeispiele (Seite 72)

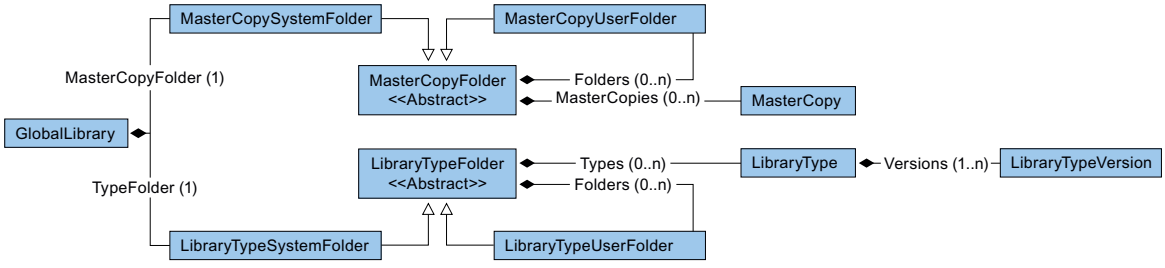
7.3 TIA Portal Openness-Objektmodell

Übersicht

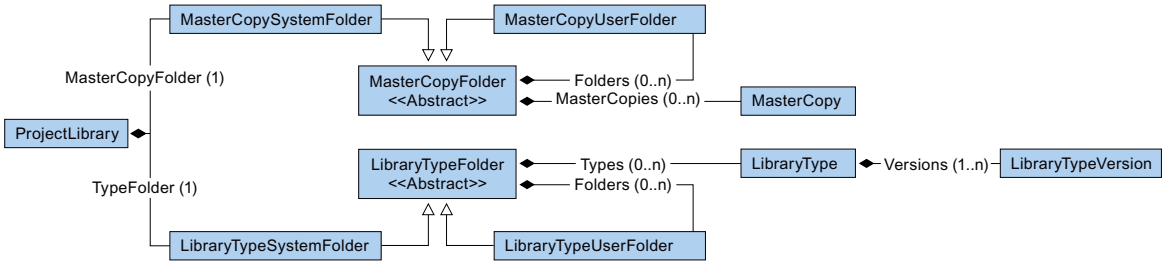
Das folgende Diagramm beschreibt die höchste Ebene des TIA Portal Openness-Objektmodells:



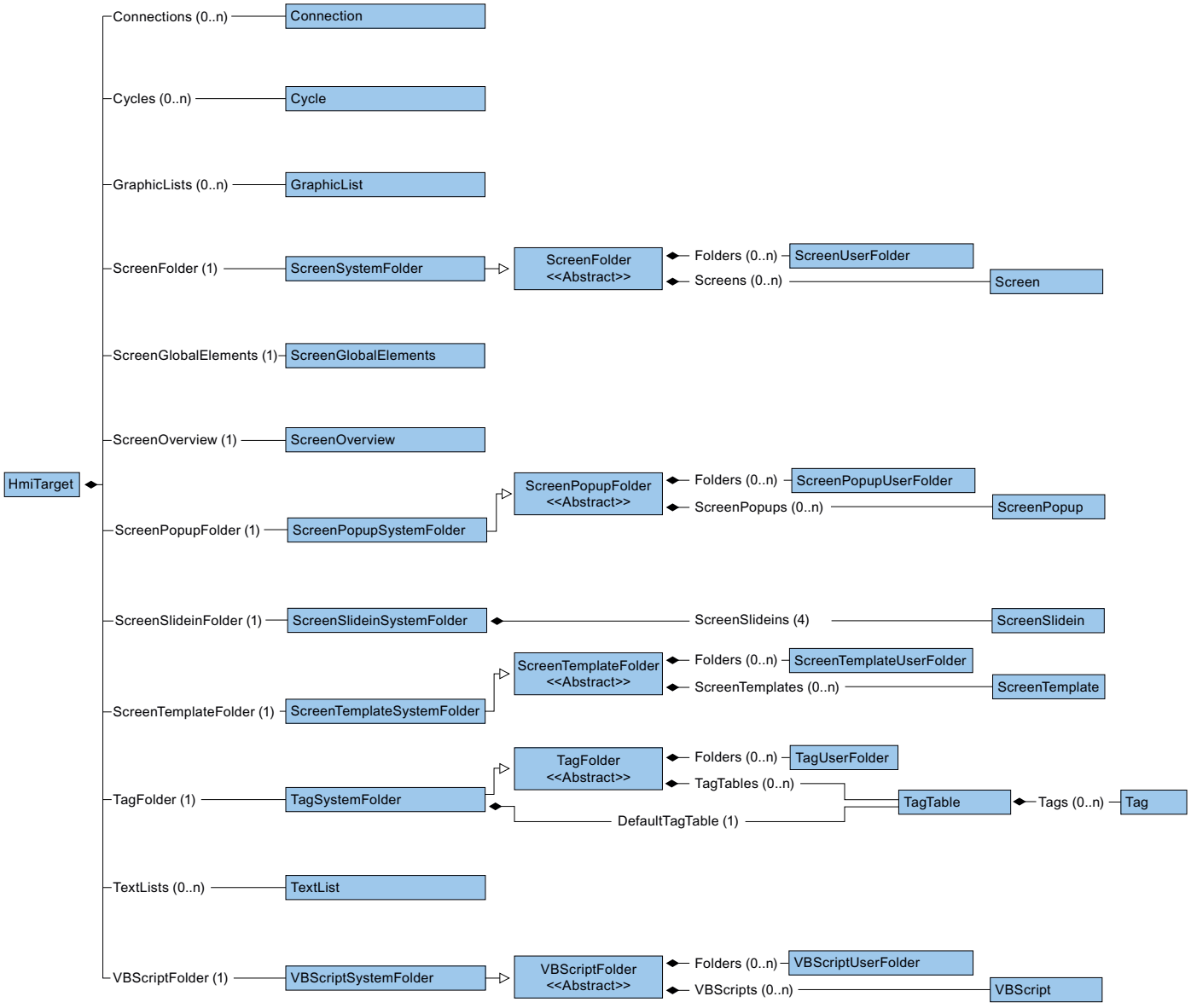
Das folgende Diagramm beschreibt die Objekte, die sich unter GlobalLibrary befinden.



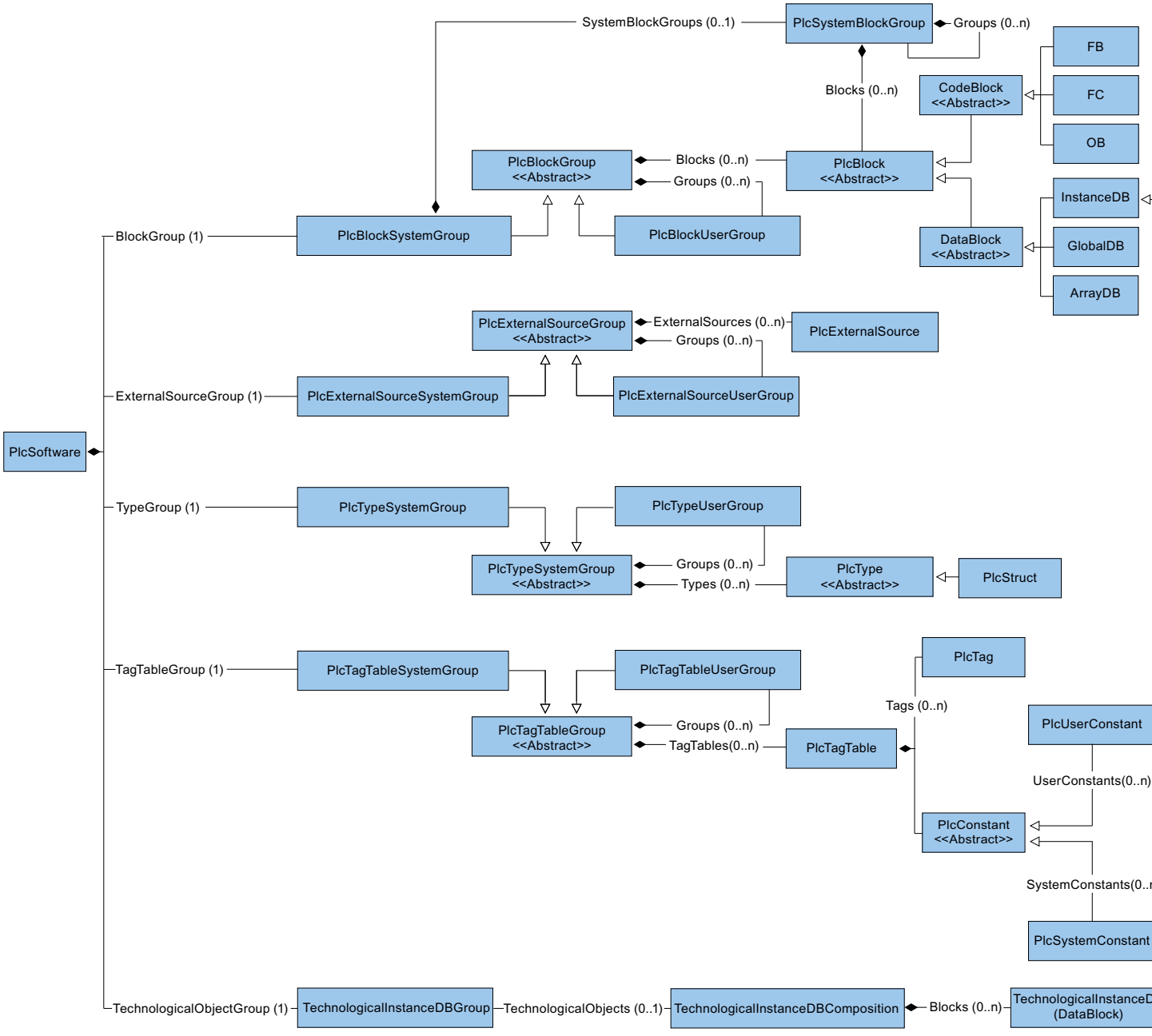
Das folgende Diagramm beschreibt die Objekte, die sich unter ProjectLibrary befinden.



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



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


SystemBlockGroups (0..1)
PlcSystemBlockGroup
Groups (0..n)
FB
FC
OB
Blocks (0..n)
CodeBlock <<Abstract>>
PlcBlockGroup <<Abstract>>
Blocks (0..n)
Groups (0..n)
PlcBlock <<Abstract>>
InstanceDB
GlobalDB
ArrayDB
DataBlock <<Abstract>>
BlockGroup (1)
PlcBlockSystemGroup
PlcBlockUserGroup
PlcExternalSourceGroup <<Abstract>>
ExternalSources (0..n)
Groups (0..n)
PlcExternalSource
ExternalSourceGroup (1)
PlcExternalSourceSystemGroup
PlcExternalSourceUserGroup
PlcSoftware
TypeGroup (1)
PlcTypeSystemGroup
PlcTypeUserGroup
PlcTypeSystemGroup <<Abstract>>
Groups (0..n)
Types (0..n)
PlcType <<Abstract>>
PlcStruct
TagTableGroup (1)
PlcTagTableSystemGroup
PlcTagTableUserGroup
PlcTag
Tags (0..n)
PlcTagTableGroup <<Abstract>>
Groups (0..n)
TagTables(0..n)
PlcTagTable
PlcUserConstant
UserConstants(0..n)
PlcConstant <<Abstract>>
SystemConstants(0..
PlcSystemConstant
TechnologicalObjectGroup (1)
TechnologicalInstanceDBGroup
TechnologicalObjects (0..1)
TechnologicalInstanceDBComposition
Blocks (0..n)
TechnologicalInstanceD
(DataBlock)

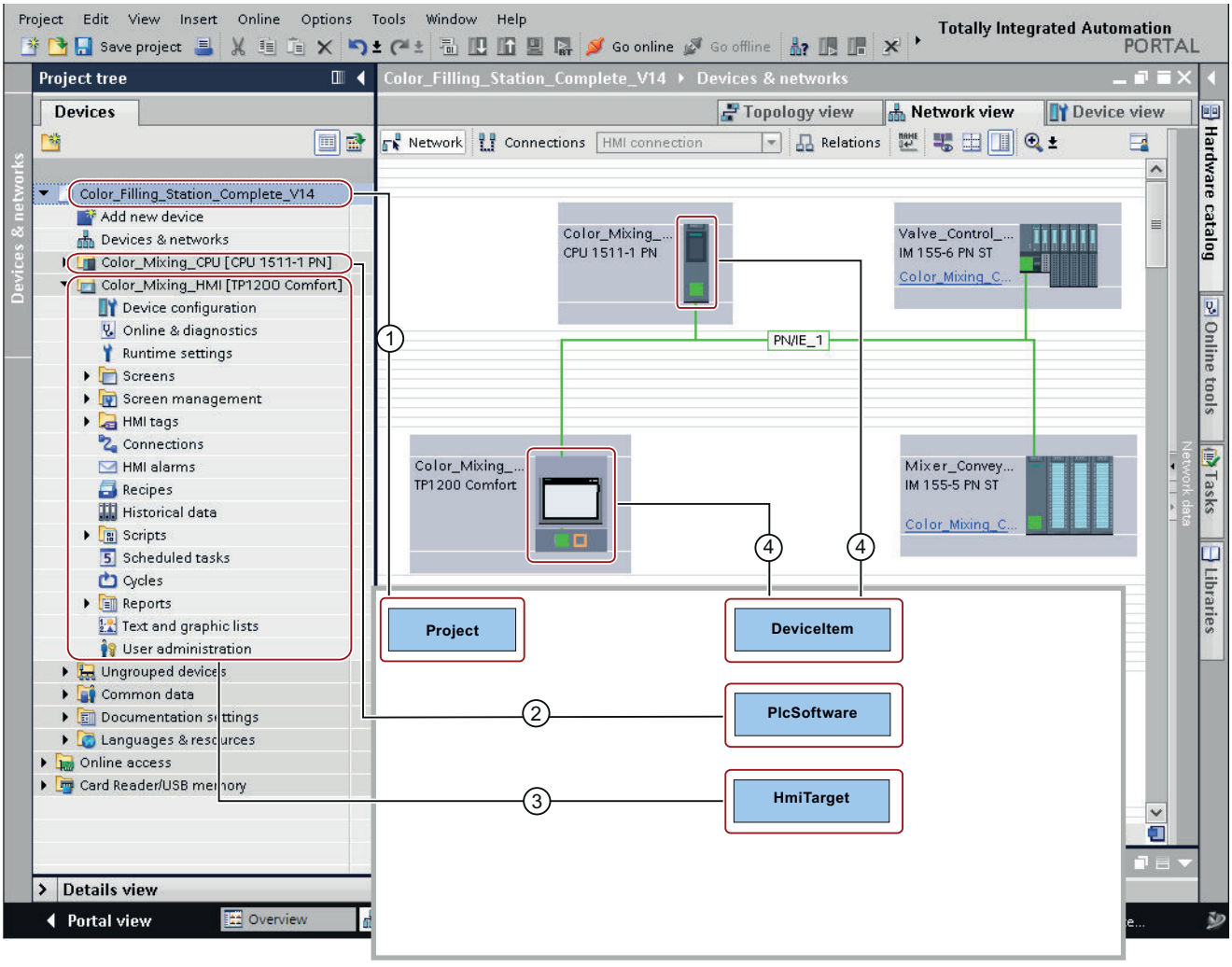
Auf Objekte in Listen zugreifen

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

- Adressieren Sie über den Index. Die Zählung innerhalb der Listen beginnt mit `0`.
- Verwenden Sie die Methode `Find`.
  Verwenden Sie diese Methode zum Adressieren eines Objekts über seinen Namen. Sie können diese Methode für eine Zusammensetzung oder Liste verwenden. Die Methode `Find` ist nicht rekursiv.
  Beispiel:
  ```
  ScreenComposition screens = folder.Screens;
  Screen screen = screens.Find("myScreen");
  ```
- Verwenden Sie symbolische Namen.

Beziehung zwischen TIA Portal und TIA Portal Openness-Objektmodell

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



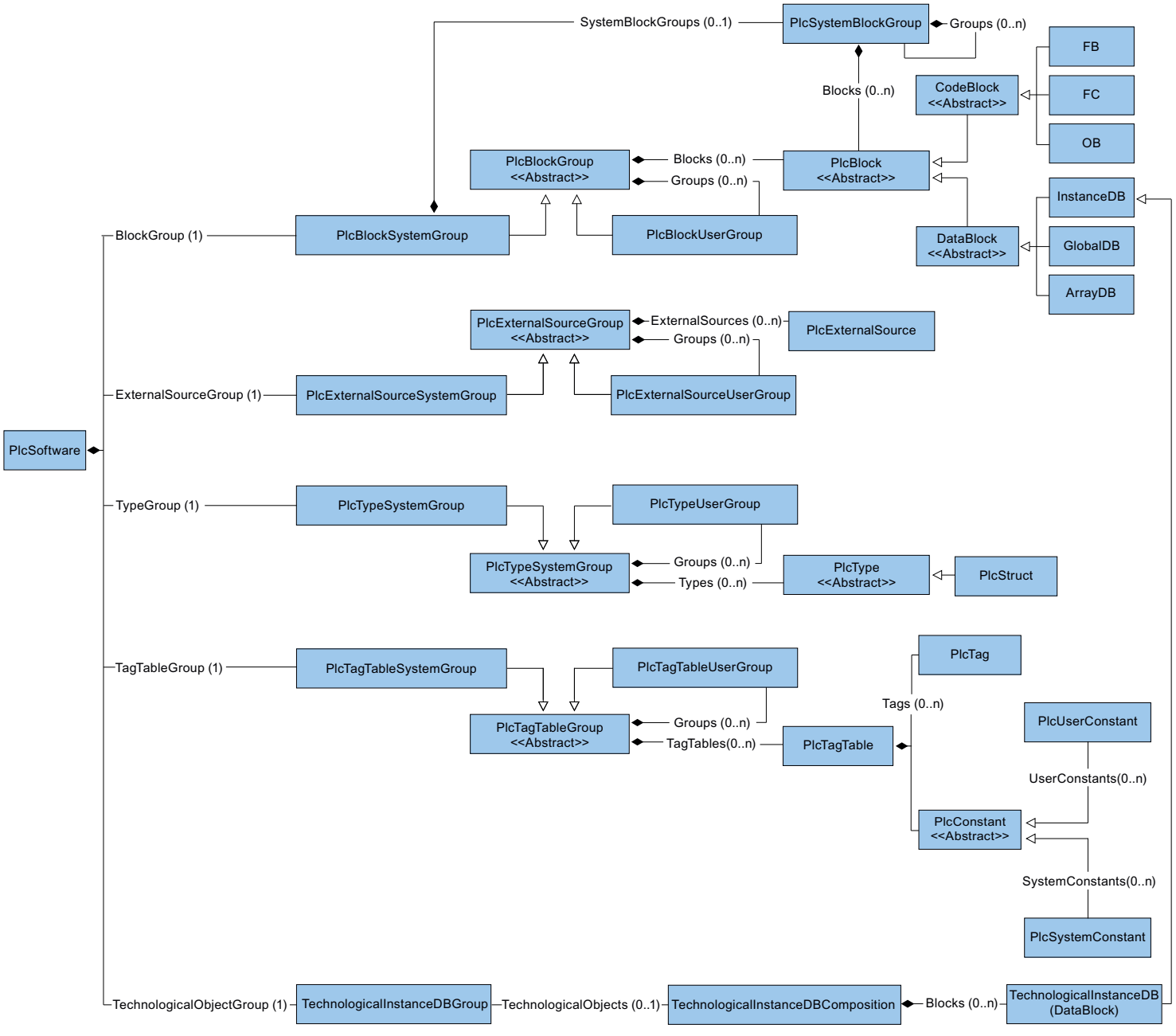
Siehe auch

Bausteine und Typen des TIA Portal Openness-Objektmodells (Seite 56)
Hierarchie von Hardware-Objekten des Objektmodells (Seite 64)

7.4 Bausteine und Typen des TIA Portal Openness-Objektmodells

Einleitung

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



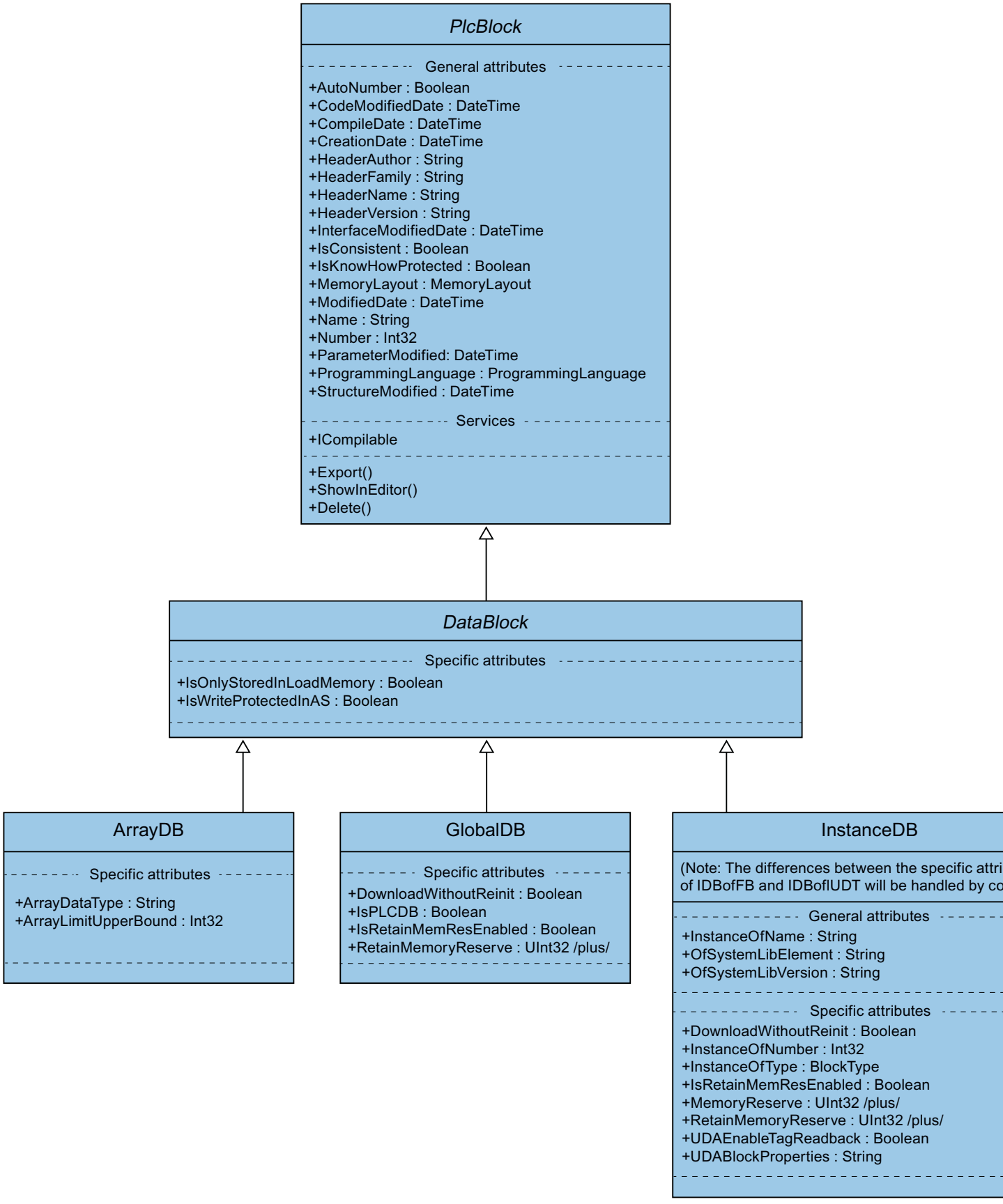
Darstellung von Bausteinen und Typen in der TIA Portal Openness API

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

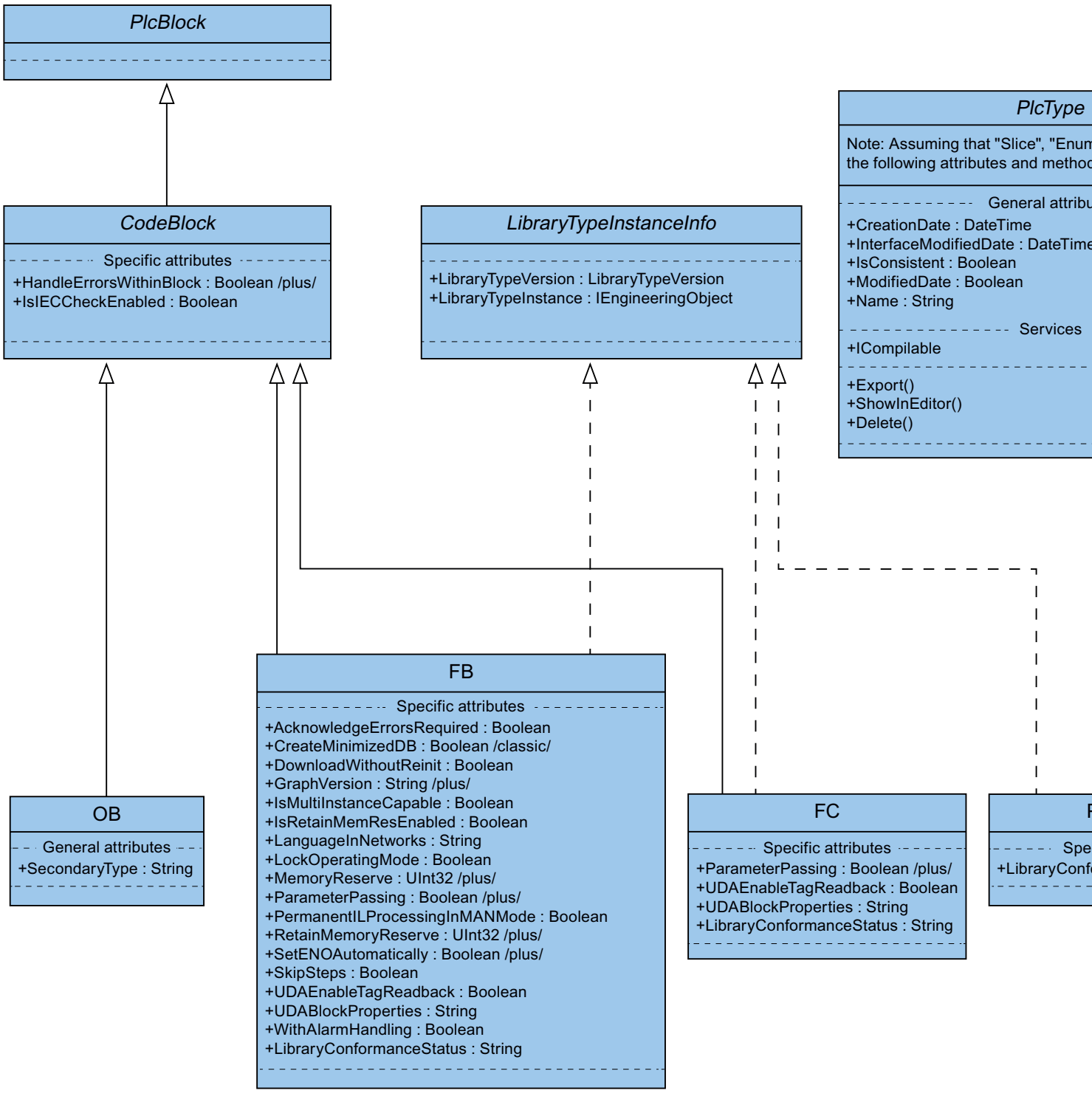
Klassendiagramme

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

Daten


PlcBlock
General attributes
+AutoNumber : Boolean
+CodeModifiedDate : DateTime
+CompileDate : DateTime
+CreationDate : DateTime
+HeaderAuthor : String
+HeaderFamily : String
+HeaderName : String
+HeaderVersion : String
+InterfaceModifiedDate : DateTime
+IsConsistent : Boolean
+IsKnowHowProtected : Boolean
+MemoryLayout : MemoryLayout
+ModifiedDate : DateTime
+Name : String
+Number : Int32
+ParameterModified: DateTime
+ProgrammingLanguage : ProgrammingLanguage
+StructureModified : DateTime
Services
+ICompilable
+Export()
+ShowInEditor()
+Delete()
DataBlock
Specific attributes
+IsOnlyStoredInLoadMemory : Boolean
+IsWriteProtectedInAS : Boolean
ArrayDB
Specific attributes
+ArrayDataType : String
+ArrayLimitUpperBound : Int32
GlobalDB
Specific attributes
+DownloadWithoutReinit : Boolean
+IsPLCDB : Boolean
+IsRetainMemResEnabled : Boolean
+RetainMemoryReserve : UInt32 /plus/
InstanceDB
(Note: The differences between the specific attri
of IDBofFB and IDBofIUDT will be handled by co
General attributes
+InstanceOfName : String
+OfSystemLibElement : String
+OfSystemLibVersion : String
Specific attributes
+DownloadWithoutReinit : Boolean
+InstanceOfNumber : Int32
+InstanceOfType : BlockType
+IsRetainMemResEnabled : Boolean
+MemoryReserve : UInt32 /plus/
+RetainMemoryReserve : UInt32 /plus/
+UDAEnableTagReadback : Boolean
+UDABlockProperties : String

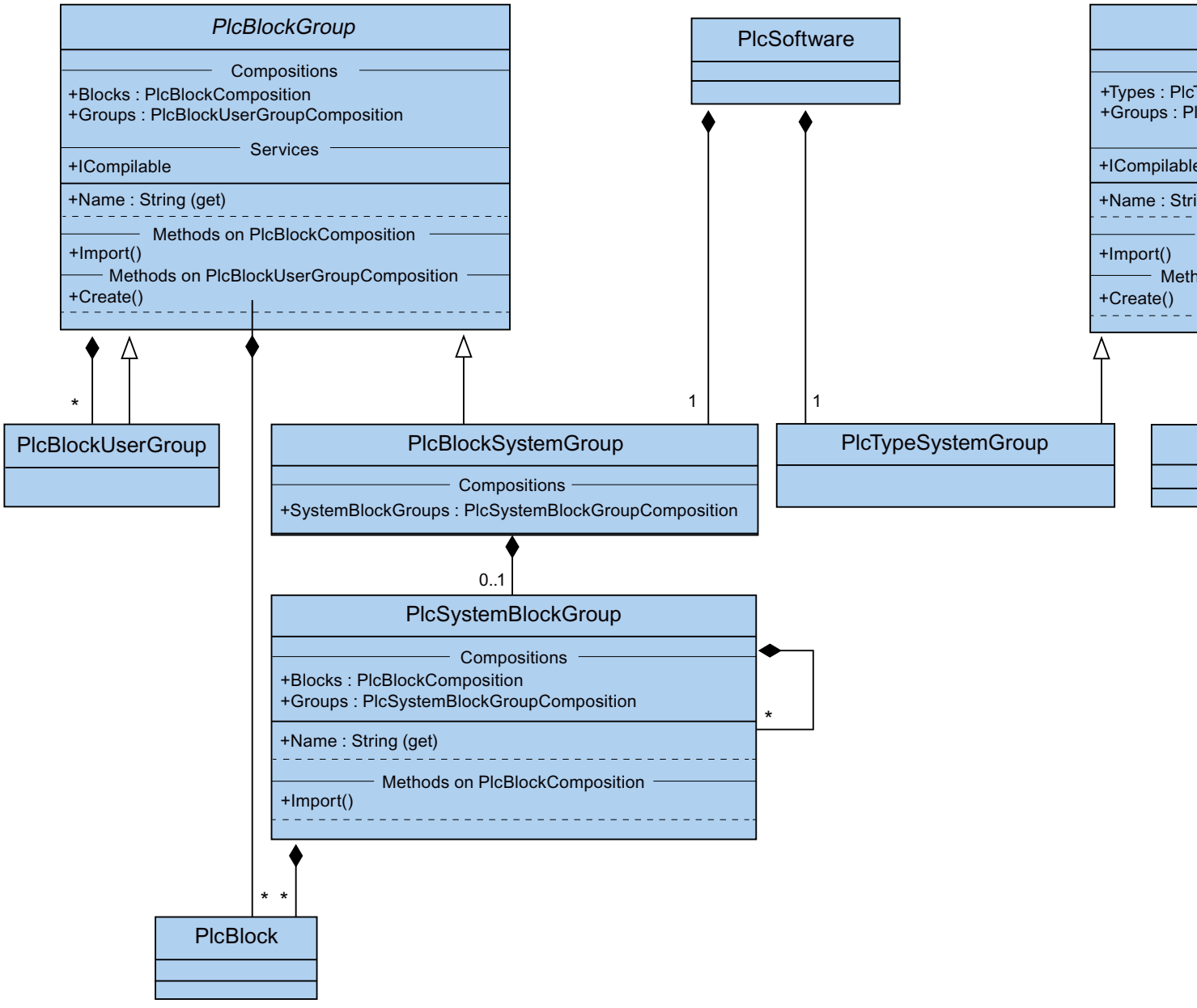
Code und Typ


PlcBlock
CodeBlock
Specific attributes
+HandleErrorsWithinBlock : Boolean /plus/
+IsIECCheckEnabled : Boolean
LibraryTypeInstanceInfo
+LibraryTypeVersion : LibraryTypeVersion
+LibraryTypeInstance : IEngineeringObject
PlcType
Note: Assuming that "Slice", "Enum
the following attributes and metho
General attribu
+CreationDate : DateTime
+InterfaceModifiedDate : DateTime
+IsConsistent : Boolean
+ModifiedDate : Boolean
+Name : String
Services
+ICompilable
+Export()
+ShowInEditor()
+Delete()
OB
General attributes
+SecondaryType : String
FB
Specific attributes
+AcknowledgeErrorsRequired : Boolean
+CreateMinimizedDB : Boolean /classic/
+DownloadWithoutReinit : Boolean
+GraphVersion : String /plus/
+IsMultiInstanceCapable : Boolean
+IsRetainMemResEnabled : Boolean
+LanguageInNetworks : String
+LockOperatingMode : Boolean
+MemoryReserve : UInt32 /plus/
+ParameterPassing : Boolean /plus/
+PermanentILProcessingInMANMode : Boolean
+RetainMemoryReserve : UInt32 /plus/
+SetENOAutomatically : Boolean /plus/
+SkipSteps : Boolean
+UDAEnableTagReadback : Boolean
+UDABlockProperties : String
+WithAlarmHandling : Boolean
+LibraryConformanceStatus : String
FC
Specific attributes
+ParameterPassing : Boolean /plus/
+UDAEnableTagReadback : Boolean
+UDABlockProperties : String
+LibraryConformanceStatus : String

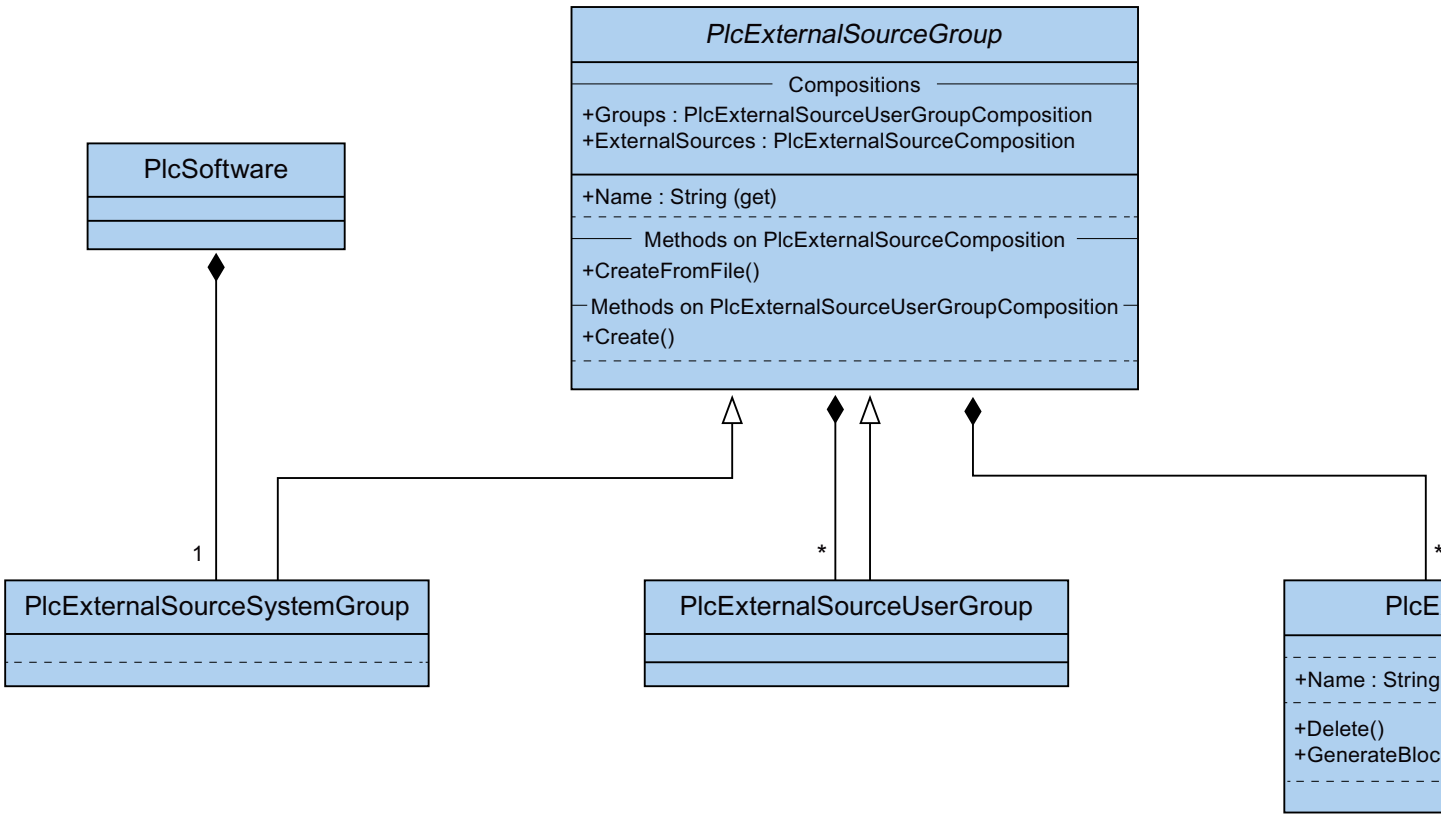
Darstellung von Gruppen für Bausteine und Typen in der TIA Portal Openness API

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

Bausteine und Typen



Externe Quellen



Siehe auch

TIA Portal Openness-Objektmodell (Seite 51)

Hierarchie von Hardware-Objekten des Objektmodells (Seite 64)

7.5 Hierarchie von Hardware-Objekten des Objektmodells

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

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

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

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

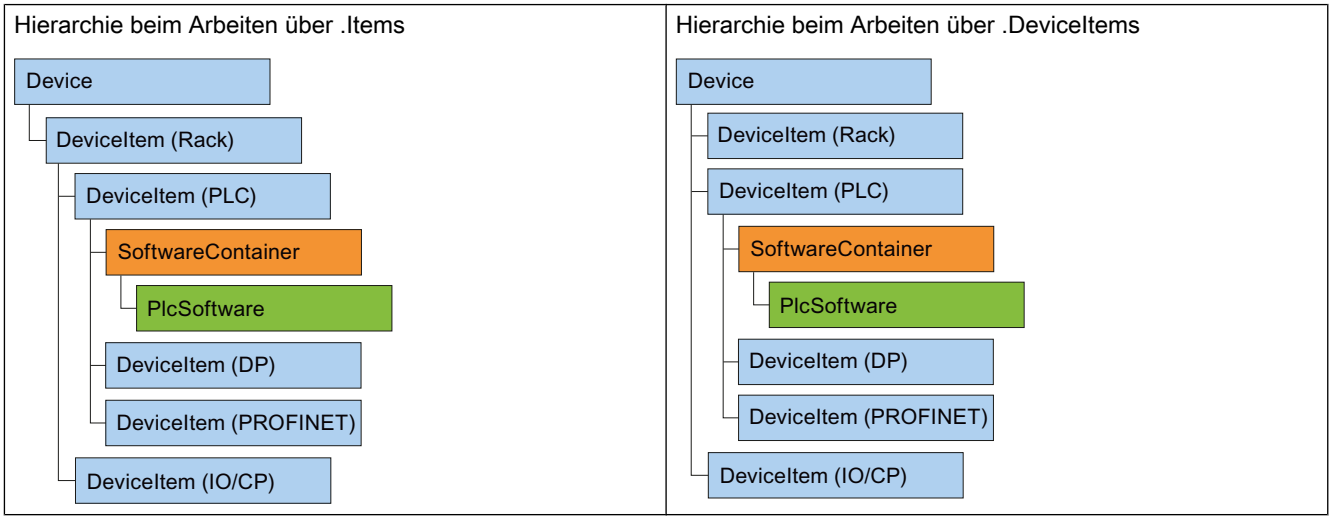
Hierbei handelt es sich um die Beziehung, die der Darstellung in Netzsicht und Gerätesicht des TIA Portals ähnelt. Das Attribut "PositionNumber" eines Geräteelements ist im Elementebereich innerhalb eines Behälters eindeutig.

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

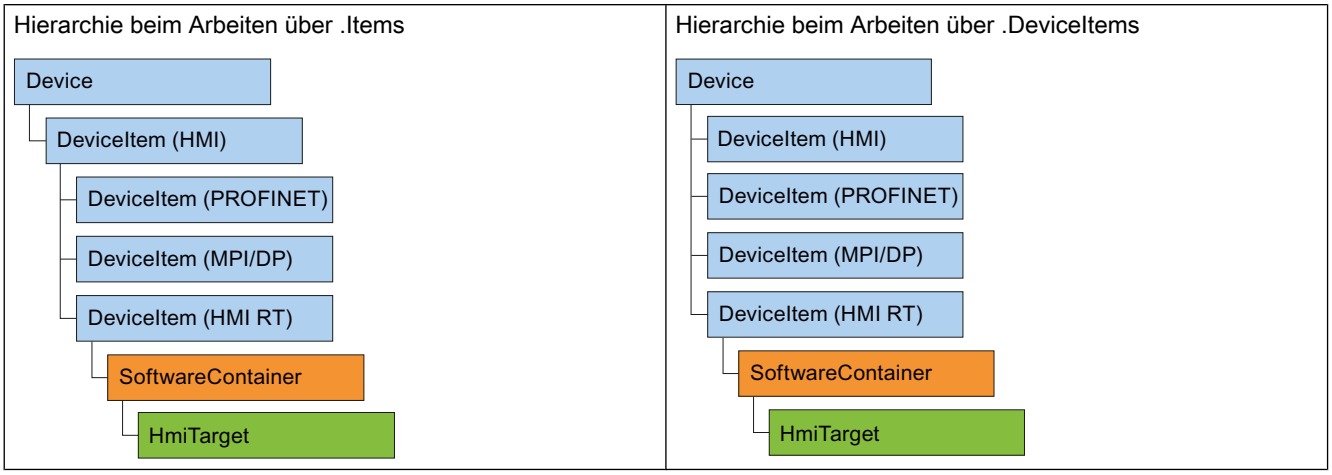
- Wenn ein Submodul als Teil eines Moduls modelliert wird (als Kind), kann das Submodul nicht ohne das Modul entfernt werden.
- Wenn Sie einem Modul ein Submodul hinzufügen und es dann vom Modul entfernen, hat dieses Kind die gleichen Eltern wie das Modul.

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

Hierarchie zwischen PLC-Geräten



Hierarchie zwischen HMI-Geräten



Siehe auch

TIA Portal Openness-Objektmodell (Seite 51)

Bausteine und Typen des TIA Portal Openness-Objektmodells (Seite 56)

7.6 Informationen über installierte TIA Portal Openness-Versionen

Voraussetzung

- TIA Portal Openness und TIA Portal sind installiert

Verwendung

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

Den Registrierungsschlüssel finden Sie unter dem folgendem Pfad:

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

Hinweis

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

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

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\Openness\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"
```

Hinweis

Wenn Sie eine Anwendungskonfigurationsdatei (Seite 74) erzeugen möchten, können Sie den Pfad für Siemens.Engineering.dll, Siemens.Engineering.Hmi.dll und Token des öffentlichen Schlüssels dem Registrierungsschlüssel entnehmen.

7.7 Beispielprogramm

Anwendungsbeispiel: API-Zugriff in einer Anwendung erstellen

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

Hinweis

Für das Anwendungsbeispiel ist eine Anwendungskonfigurationsdatei (Seite 74) erforderlich.

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

Vorgehensweise in Einzelschritten

1. TIA Portal in der Entwicklungsumgebung bekannt machen

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

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

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

Hinweis

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

2. Namensraum für das TIA Portal veröffentlichen

Fügen Sie den folgenden Code hinzu:

```
using Siemens.Engineering;
```

3. TIA Portal veröffentlichen und starten

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

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

4. Projekt öffnen

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

```
ProjectComposition projects = tiaPortal.Projects;
Console.WriteLine("Opening Project...");
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. Geräte eines Projekts enumerieren

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

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

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

6. Projekt speichern und schließen

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

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

7.8 Einsatz der Codebeispiele

Aufbau der Codeausschnitte

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

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

Zum Ausführen dieses Codeausschnitts benötigen Sie Folgendes:

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

Hinweis

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

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

Beispiel für Ausführung des Codeausschnitts

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

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

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

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

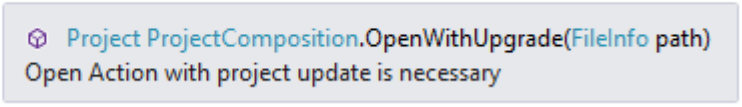
7.9 Allgemeine Funktionen

7.9.1 Unterstützung von IntelliSense durch TIA Portal Openness

Verwendung

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

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



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

7.9.2 Verbindung zum TIA Portal aufbauen

Einleitung

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

Hinweis

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

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

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

Hinweis

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

Starten des TIA Portals mit einer Anwendungskonfigurationsdatei

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

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

Für die Anwendungskonfigurationsdatei können Sie den folgenden Codeausschnitt verwenden:

```
<?xml version="1.0"?>
<configuration>
    <runtime>
        <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
            <dependentAssembly>
                <assemblyIdentity name="Siemens.Engineering" culture="neutral"
publicKeyToken="d29ec89bac048f84"/>
                <!-- Edit the following path according to your installation -->
              <codeBase version="15.1.0.0" href="FILE://C:\Program Files\Siemens\Automation
\Portal V15.1\PublicAPI\V15.1\Siemens.Engineering.dll"/>
            </dependentAssembly>
            <dependentAssembly>
                <assemblyIdentity name="Siemens.Engineering.Hmi" culture="neutral"
publicKeyToken="d29ec89bac048f84"/>
                <!-- Edit the following path according to your installation -->
              <codeBase version="15.1.0.0" href="FILE://C:\Program Files\Siemens\Automation
\Portal V15.1\PublicAPI\V15.1\Siemens.Engineering.Hmi.dll"/>
            </dependentAssembly>
        </assemblyBinding>
    </runtime>
</configuration>
```

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

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

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

Starten des TIA Portals mit der Methode "AssemblyResolve"

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

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

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

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

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

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

Auf aktive Instanzen des TIA Portals zugreifen

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

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

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

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

- Verbindung einmal erlauben
- Verbindung nicht erlauben
- Verbindungen von dieser Applikation immer erlauben
  Weitere Informationen hierzu siehe TIA Portal Openness-Firewall (Seite 79).

Hinweis

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

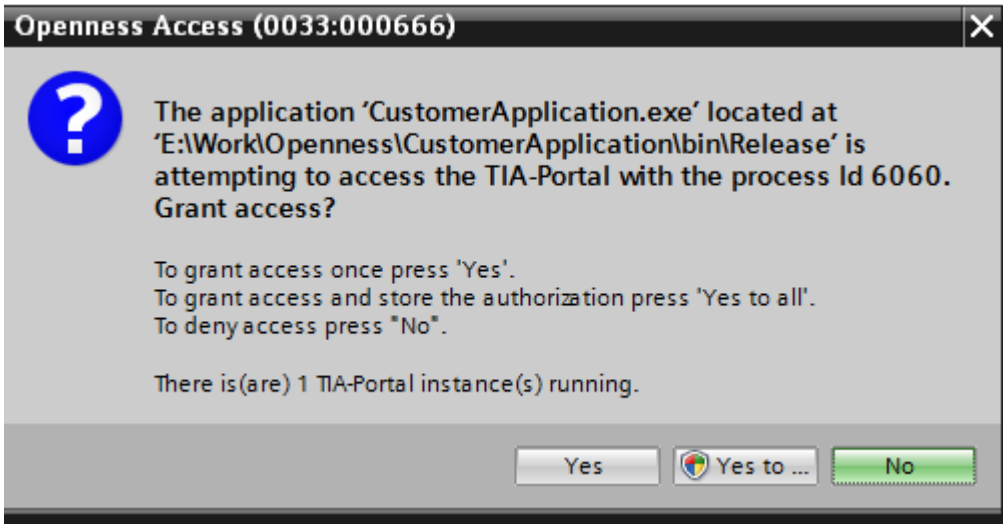
Nach Herstellung der Verbindung zum Prozess können Sie mit Hilfe eines der folgenden Attribute Informationen zu den Instanzen des TIA Portals abrufen:

Attribut	Information
`InstalledSoftware as IList<TiaPortalProduct>`	Gibt Informationen über die installierten Produkte zurück.
`Mode as TiaPortalMode`	Gibt den Modus zurück, in dem das TIA Portal gestartet wurde (WithoutUserInterface/WithUserInterface).
`AttachedSessions as IList<TiaPortalSession>`	Gibt ein Liste von Anwendungen zurück, die mit dem TIA Portal verbunden sind.
`ProjectPath as FileInfo`	Gibt den Dateinamen des im TIA Portal geöffneten Projekts einschließlich des Ordners zurück, z. B. `"D:\WinCCProjects\ColorMixing\ColorMixing.ap14"` Wenn kein Projekt geöffnet ist, wird eine leere Zeichenfolge zurückgegeben.
`ID as int`	Gibt die Prozess-ID der Instanz des TIA Portals zurück.
`Path as FileInfo`	Gibt den Pfad zur ausführbaren Datei des TIA Portals zurück.

7.9.3 TIA Portal Openness-Firewall

Eingabeaufforderung der TIA Portal Openness-Firewall

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



Verbindung zum TIA Portal einmal erlauben

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

Whitelist-Eintrag durch Verbinden mit dem TIA Portal hinzufügen

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

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

Whitelist-Eintrag ohne TIA Portal hinzufügen

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

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\Openness\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="
```

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

```
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 Event-Handler

Ereignis-Handler in der TIA Portal Openness-Anwendung

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

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

Programmcode

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

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

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

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

private static void TiaPortal_Notification(object sender, NotificationEventArgs e)

{
    ....
}

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

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

Attribute von Benachrichtigungen des TIA Portals

Benachrichtigungen des TIA Portals haben die folgenden Attribute:

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

Attribute von Bestätigungen

Bestätigungen haben die folgenden Attribute:

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

Siehe auch

Dialoge mit Systemmeldungen programmgesteuert bestätigen (Seite 82)

7.9.5 Dialoge mit Systemmeldungen programmgesteuert bestätigen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Ereignis-Handler sind registriert.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)

Verwendung

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

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

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

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

- `OK`
- `Retry`
- `Yes`
- `YesToAll`

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

Programmcode

Ändern Sie den folgenden Programmcode, um auf ein Bestätigungsereignis zu reagieren:

```
...
    tiaPortal.Confirmation += TiaPortalConfirmation;
...
private void TiaPortalConfirmation(object sender, ConfirmationEventArgs e)
{
    ...
}
```

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

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

7.9.6 Verbindung zum TIA Portal beenden

Einleitung

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

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

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

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

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

Hinweis

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

Programmcode

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

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

Siehe auch

Event-Handler (Seite 80)

7.9.7 Diagnoseschnittstellen im TIA Portal

Verwendung

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

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

Klasse TiaPortalProcess

Mitglied	Typ	Funktion
AcquisitionTime	DateTime	Zeitpunkt, zu dem das TiaPortal-Process-Objekt erfasst wurde. Da das Objekt TiaPortalProcess eine vollkommen statische Momentaufnahme vom Zustand des TIA Portals zu einem gegebenen Zeitpunkt darstellt, können darin enthaltene Informationen veraltet sein.
Attach	TiaPortal	Hängt sich dem gegebenen TiaPortalProcess an und gibt eine Instanz des TIA Portals zurück.
AttachedSessions	IList<TiaPortalSession>	Eine Sammlung aller anderen Sitzungen, die gegenwärtig an dasselbe TIA Portal angehängt sind. Diese Sammlung kann leer sein. Jede Sitzung wird durch ein nachstehend beschriebenes TiaPortalSession-Objekt repräsentiert.
Attaching	EventHandler<AttachingEventArgs>	Dieses Ereignis ermöglicht einer Anwendung, Versuche zum Anhängen an das TIA Portal zuzulassen. Wenn eine andere Anwendung versucht, sich dem TIA Portal anzuhängen, werden die Teilnehmer dieses Ereignisses benachrichtigt. Die Teilnehmer haben dann 10 Sekunden Zeit, um dieser Anbindung zuzustimmen. Wenn ein Teilnehmer dieses Ereignis ignoriert oder nicht rechtzeitig reagiert, gilt das als Ablehnung und der anderen Anwendung wird das Anhängen nicht erlaubt. Abgestürzte Anwendungen, die nicht in der Lage sind, auf dieses Ereignis zu reagieren, können einer Anwendung die Erlaubnis zum Anhängen nicht verweigern.
Dispose	void	Schließt die betreffende Instanz des TIA Portals.
Id	int	Die Prozess-ID des TIA Portals.
InstalledSoftware	IList<TiaPortalProduct>	Eine Sammlung aller gegenwärtig installierten Produkte als Teil des TIA Portals. Jedes Produkt wird durch ein nachstehend beschriebenes TiaPortalProduct-Objekt dargestellt.

Mitglied	Typ	Funktion
Mode	TiaPortalMode	Gibt den Modus zurück, in dem das TIA Portal gestartet wurde. Die aktuellen Werte sind WithUserInterface und WithoutUserInterface.
Path	FileInfo	Der Pfad zur ausführbaren Datei des TIA Portals.
ProjectPath	FileInfo	Der Pfad zu dem Projekt, das gegenwärtig im TIA Portal geöffnet ist. Wenn kein Projekt geöffnet ist, hat dieses Attribut den Wert Null.

Klasse TiaPortalSession

Mitglied	Typ	Funktion
AccessLevel	TiaPortalAccessLevel	Zugriffsebene der Sitzung. Wird dargestellt als Merker-Enumeration, wobei mehrere Zugriffsebenen möglich sind. TiaPortalAccessLevel wird nachstehend ausführlich beschrieben.
AttachTime	DateTime	Der Zeitpunkt, zu dem die Verbindung mit dem TIA Portal hergestellt wurde.
Id	int	Die ID der aktuellen Sitzung.
IsActive	bool	Gibt "wahr" zurück, wenn das TIA Portal gegenwärtig einen Aufruf aus dieser Sitzung verarbeitet.
Dispose	void	Trennt die Verbindung des Prozesses zum TIA Portal. Diese Methode erzwingt nicht die Beendigung des Prozesses selbst, so wie es mit System.Diagnostics.Process.Kill der Fall wäre. Die Anwendung, deren Verbindung beendet wird, erhält dennoch ein Disposed-Ereignis. Es erfolgt aber kein weiterer Hinweis darauf, weshalb die Verbindung beendet wurde.
ProcessId	int	Die Prozess-ID des angehängten Prozesses.
ProcessPath	FileInfo	Der Pfad der ausführbaren Datei des angehängten Prozesses.

Mitglied	Typ	Funktion
TrustAuthority	TiaPortalTrustAuthority	Gibt an, ob die aktuelle Sitzung von einem signierten Prozess gestartet wurde und ob es ein TIA Portal Openness-Zertifikat ist oder nicht. TrustAuthority ist eine Merker-Enumeration und wird nachstehend beschrieben.
UtilizationTime	TimeSpan	Zeitraum, den der Prozess aktiv im TIA Portal verbracht hat. Kombiniert mit dem Attribut AttachTime kann dies dazu genutzt werden, prozentuale Nutzungsdaueranteile oder ähnliche Daten zu ermitteln.
Version	string	Die Version der Siemens.Engineering.dll, der die Sitzung angehängt wird.

Enum TiaPortalAccessLevel

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

Enum TiaPortalTrustAuthority

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

Klasse TiaPortalProduct

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

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

```
public void TiaPortalDiagnostics()
{
    IList<TiaPortalProcess> tiaPortalProcesses = TiaPortal.GetProcesses();
    foreach (TiaPortalProcess tiaPortalProcess in tiaPortalProcesses)
    {
        Console.WriteLine("Process ID: {0}", tiaPortalProcess.Id);
        Console.WriteLine("Path: {0}", tiaPortalProcess.Path);
        Console.WriteLine("Project: {0}", tiaPortalProcess.ProjectPath);
        Console.WriteLine("Timestamp: {0}", tiaPortalProcess.AcquisitionTime);
        Console.WriteLine("UI Mode: {0}", tiaPortalProcess.Mode);
        //See method body below.
        Console.WriteLine("Installed Software:");
        EnumerateInstalledProducts(tiaPortalProcess.InstalledSoftware);
        Console.WriteLine("Attached Openness Applications:");
        foreach (TiaPortalSession session in tiaPortalProcess.AttachedSessions)
        {
            Console.WriteLine("Process: {0}", session.ProcessPath);
            Console.WriteLine("Process ID: {0}", session.ProcessId);
            DateTime attachTime = session.AttachTime;
            TimeSpan timeSpentAttached = DateTime.Now - attachTime;
            TimeSpan utilizationTime = session.UtilizationTime;
            long percentageTimeUsed = (utilizationTime.Ticks / timeSpentAttached.Ticks) *
100;
            Console.WriteLine("AttachTime: {0}", attachTime);
            Console.WriteLine("Utilization Time: {0}", utilizationTime);
            Console.WriteLine("Time spent attached: {0}", timeSpentAttached);
            Console.WriteLine("Percentage of attached time spent using TIA Portal: {0}",
percentageTimeUsed);
            Console.WriteLine("AccessLevel: {0}", session.AccessLevel);
            Console.WriteLine("TrustAuthority: {0}", session.TrustAuthority);
            if ((session.TrustAuthority & TiaPortalTrustAuthority.Certified) !=
TiaPortalTrustAuthority.Certified)
            {
                Console.WriteLine("TrustAuthority doesn't match required level, attempting
to terminate connection to TIA Portal."); session.Dispose();
            }
        }
    }
}
public void EnumerateInstalledProducts(IEnumerable<TiaPortalProduct> products)
{
    foreach (TiaPortalProduct product in products)
    {
        Console.WriteLine("Name: {0}", product.Name);
        Console.WriteLine("Version: {0}", product.Version);
        //recursively enumerate all option packages
        Console.WriteLine("Option Packages \n:");
        EnumerateInstalledProducts(product.Options);
    }
}
```

Sicherheitsrelevante Information

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

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

7.9.8 Exclusive access

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

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

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

Hinweis

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

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

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

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

- der Assembly-Titel aus den Manifestdaten, wenn verfügbar, andernfalls der Prozessname
- die Prozess-ID
- die SID

Hinweis

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

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

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

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

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

7.9.9 Transaktionsbehandlung

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Operation

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

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

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

Hinweis

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

Konsistente Umsetzung oder Rückgängigmachung

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

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

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

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

Einschränkungen

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

- Compile
- Go Online
- Go Offline
- ProjectText Import
- ProjectText Export
- Open Global Library
- Close Global Library
- Project Create
- Project Open
- Project OpenWithUpgrade
- Project Save
- Project Close

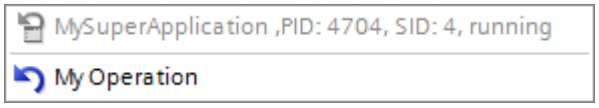
Undo-Verhalten

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

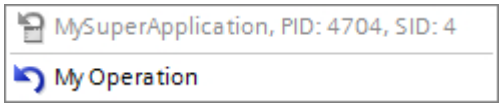
- der Assembly-Titel aus den Manifestdaten, wenn verfügbar, andernfalls der Prozessname
- die Prozess-ID
- die SID
- optional ein Hinweis darauf, dass der Client-Prozess noch läuft

Diese Einträge gehören zu einer der folgenden beiden Arten:

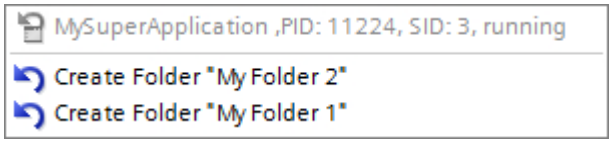
1. Die Operationen, die als Resultat der Verwendung einer "Transaction" in einer Undo-Transaktion zusammengefasst werden, haben die Beschreibung wie von der Client-Anwendung bereitgestellt, als die "Transaction" instanziiert wurde.
   - Undo-Eintrag für eine laufende Client-Anwendung:

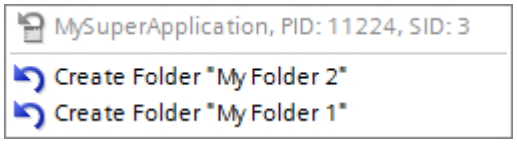
   - Undo-Eintrag für eine gestoppte Client-Anwendung:

2. Für die Operationen, die individuell ausgeführt werden, existieren individuelle Undo-Einträge zur Beschreibung der Operation, wie im entsprechenden Metadatenbefehl definiert.
   - Undo-Eintrag für eine laufende Client-Anwendung:

   - Undo-Eintrag für eine gestoppte Client-Anwendung:

7.9.10 Ein Objekt DirectoryInfo/FileInfo erstellen

Anwendung

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

Programmcode

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

```
..
    //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 Unterstützung der Selbstbeschreibung für Attribute, Navigatoren, Aktionen und Dienste

Verwendung

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

Unterstützung der Selbstbeschreibung für IEngineeringObject

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

Unterstützung der Selbstbeschreibung für IEngineeringServiceProvider

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

Klasse EngineeringCompositionInfo

Attributname	Rückgabewerte
Name	Name der Zusammensetzung

Klasse EngineeringAttributeInfo

Attributname	Rückgabewerte
AccessMode	Die vom Attribut unterstützte Zugriffsebene. Dieses Attribut ist kombinierbar und wird nachstehend ausführlich beschrieben.
Name	Name des Attributs.

Klasse EngineeringInvocationInfo

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

Klasse EngineeringServiceInfo

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

Enum AccessMode

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

Klasse EngineeringInvocationParameterInfo

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

Programmcode

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

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

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

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

7.10 Funktionen der Projekte und Projektdaten

7.10.1 Projekt öffnen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Das zu öffnende Projekt ist in keiner anderen Instanz des TIA Portals geöffnet.

Hinweis

Rückgängigmachen eines Projekt-Upgrades

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

Verwendung

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

Die Methode `Projects.Open` greift ausschließlich auf Projekte zu, die mit der aktuellen Version des TIA Portal s angelegt oder auf die aktuelle Version hochgerüstet wurden. Wenn Sie mit der Methode `Projects.Open` auf ein Projekt einer Vorgängerversion zugreifen, wird eine Ausnahme zurückgegeben. Verwenden Sie die Methode `OpenWithUpgrade` zum Öffnen von Projekten, die mit älteren Versionen von TIA Portal angelegt wurden.

Hinweis

Kein Zugriff auf schreibgeschützte Projekte

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

Programmcode

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

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

Öffnen eines UMAC-geschützten Projekts

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

Programmcode

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

Mehrere Projekte öffnen

Sie können in einer Instanz des TIA Portals mehrere Projekte öffnen. Sie müssen dann entscheiden, ob ein Projekt als primäres oder sekundäres Projekt geöffnet werden soll. Wird ein Projekt als primäres Projekt geöffnet, wird es in der Projektnavigation angezeigt, wenn die Openness-Anwendung mit einem TIA Portal verknüpft ist. Wird ein Projekt als sekundäres Projekt geöffnet, wird es in der Benutzeroberfläche nicht angezeigt. Das UMAC-geschützte Projekt kann jedoch immer als sekundäres Projekt geöffnet werden, allerdings nur mit Schreibschutz, auch wenn Lese- und Schreibrechte vorhanden sind. Es ist nicht notwendig, dass ein primäres Projekt geöffnet ist, damit ein sekundäres Projekt geöffnet werden kann.

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

Programmcode

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

Mit älteren Versionen angelegte Projekte öffnen

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

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

Hinweis

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

Programmcode

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

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

Programmcode für ein UMAC-geschütztes Projekt

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

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

7.10.2 Projekt anlegen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)

Verwendung

Projekte können über die TIA Portal Openness API erstellt werden

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

ProjectComposition.Create

Ändern Sie folgenden Programmcode:

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

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

Entsprechend diesem Beispiel

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

Hinweis

Parameter targetDirectory

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

IEngineeringComposition.Create

Ändern Sie folgenden Programmcode:

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

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

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

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

Entsprechend diesem Beispiel

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

Parameter zum Erstellen eines Projekts mit optionalen Projektattributen

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

7.10.3 Auf allgemeine Einstellungen des TIA Portals zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

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

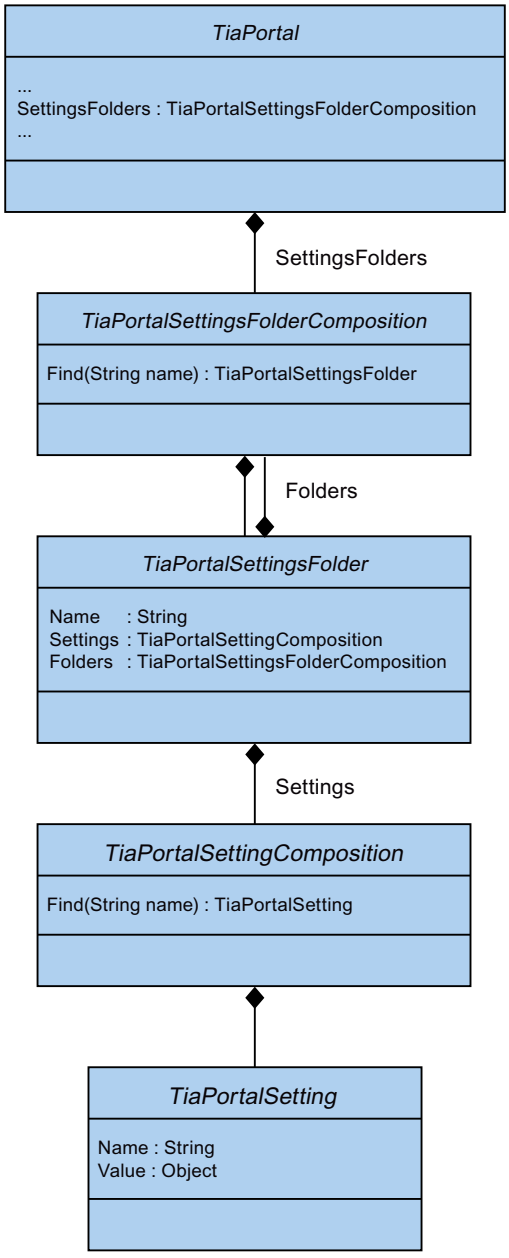
- Aktuelle Sprache der Benutzeroberfläche
- Option "In Projekt suchen", um den Suchindex zu erstellen, der für die Suche in einem Projekt benötigt wird.

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

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

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

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


TiaPortal
...
SettingsFolders : TiaPortalSettingsFolderComposition
...
SettingsFolders
TiaPortalSettingsFolderComposition
Find(String name) : TiaPortalSettingsFolder
Folders
TiaPortalSettingsFolder
Name : String
Settings : TiaPortalSettingComposition
Folders : TiaPortalSettingsFolderComposition
Settings
TiaPortalSettingComposition
Find(String name) : TiaPortalSetting
TiaPortalSetting
Name : String
Value : Object

Programmcode: Im Projekt suchen

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

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

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

Programmcode: Sprache der Benutzeroberfläche

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

```
private static void SetUILanguage(Project project)
{

    TiaPortalSettingsFolder generalSettingsFolder =
tiaPortal.SettingsFolders.Find("General");

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

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

}
```

Siehe auch

Hierarchie von Hardware-Objekten des Objektmodells (Seite 64)

7.10.4 Schreibgeschütztes TIA Portal-Projekt öffnen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Anwendung

Mithilfe von TIA Portal Openness können auch bei einem schreibgeschützten TIA Portal-Projekt bestimmte Vorgänge ausgeführt werden. Sie können auf ein schreibgeschütztes Projekt zugreifen, können jedoch nicht den vollständigen Funktionsumfang nutzen, der einem Benutzer mit Lese-/Schreibzugriff zur Verfügung steht. Beispielsweise kann ein Benutzer, der nur über Leserechte verfügt, mit Openness ein UMAC-geschütztes Projekt wie in Öffnen eines Projekts (Seite 99) beschrieben öffnen. Die Nutzbarkeit dieser Funktion gilt jedoch nicht für Referenzprojekte.

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

Integrierte Funktionen

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

Aktivierte nicht-verändernde Aktionen

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

Siehe auch

Projekt öffnen (Seite 99)

7.10.5 Auf Sprachen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

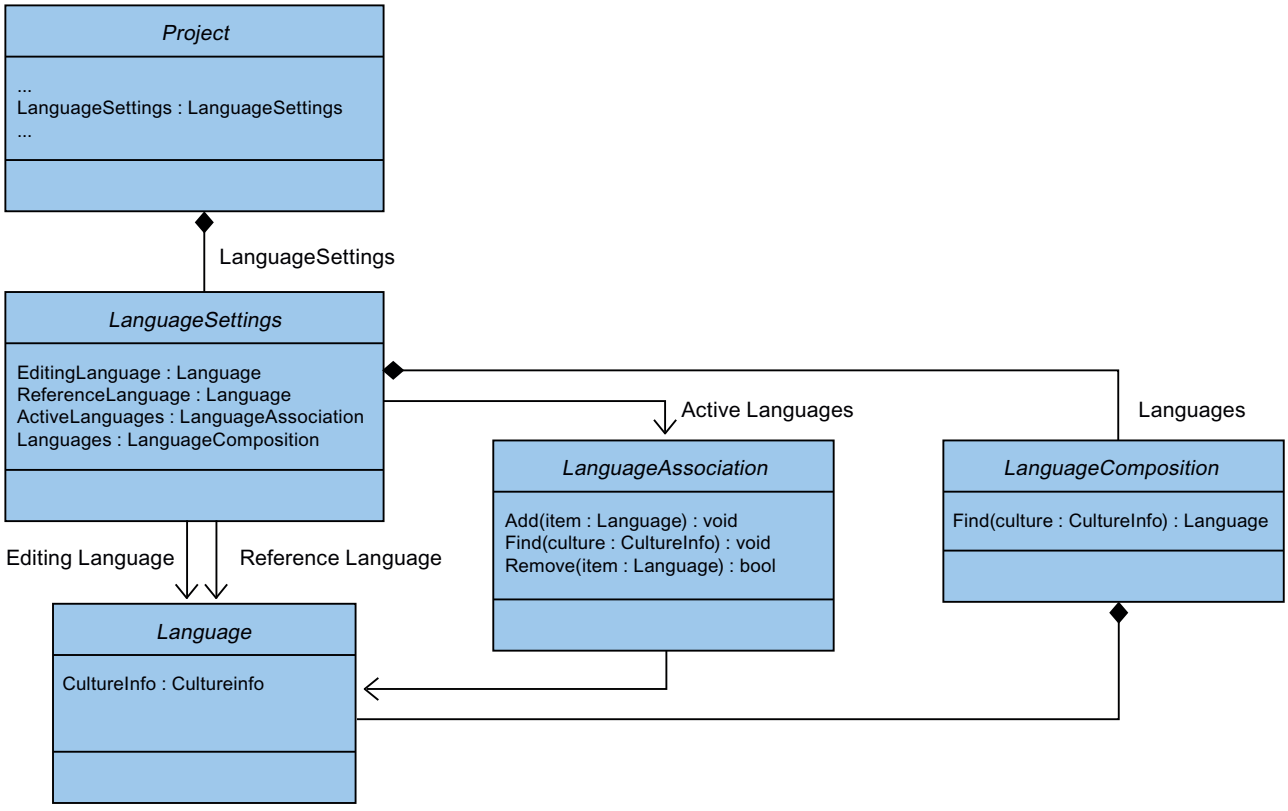
Anwendung

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

TIA Portal Openness unterstützt den folgenden Zugriff auf die Projektsprachen:

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

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



Programmcode: Sprachen festlegen

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

```
Project project = ...;

    LanguageSettings languageSettings = project.LanguageSettings;

    LanguageComposition supportedLanguages = languageSettings.Languages;
    LanguageAssociation activeLanguages = languageSettings.ActiveLanguages;

    Language supportedGermanLanguage =
supportedLanguages.Find(CultureInfo.GetCultureInfo("de-DE"));
    activeLanguages.Add(supportedGermanLanguage);

    languageSettings.EditingLanguage = supportedGermanLanguage;
    languageSettings.ReferenceLanguage = supportedGermanLanguage;
```

Programmcode: Eine aktive Sprache deaktivieren

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

```
Project project = ...;

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

Siehe auch

Hierarchie von Hardware-Objekten des Objektmodells (Seite 64)

7.10.6 Objektstruktur und -Attribute ermitteln

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

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

- Untergeordnete Objekte
- Kindzusammensetzungen
- Alle Attribute

Signatur

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

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

Programmcode: Objekte oder Zusammensetzungen ermitteln

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

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

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

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

Programmcode: Attribute ermitteln

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

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

7.10.7 Auf Software-Ziel zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

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

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

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

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

7.10.8 Auf mehrsprachige Texte zugreifen und sie enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

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

`MultilingualTextItemComposition` unterstützt die folgende `Find`-Methode:

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

Jedes `MultilingualTextItem` hat die folgenden Attribute:

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

Programmcode: Mehrsprachige Texte festlegen

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

Programmcode: Mehrsprachige Texte für Geräte festlegen

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

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

7.10.9 Projektbezogene Attribute lesen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

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

Projektattribute

Die Projektattribute liefern die folgenden Informationen:

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

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

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

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

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

Ändern Sie den folgenden Programmcode, um Kommentartext zu erhalten:

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

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

Projekthistorie

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

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

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

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

Hinweis

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

Verwendete Produkte

Das Objekt `UsedProduct` enthält die folgenden Informationen:

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

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

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

7.10.10 Projektgrafik löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

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

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

7.10.11 Projekt übersetzen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Alle Geräte sind "offline".

Anwendung

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

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

Hinweis

Format des Zeitstempels

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

Signatur

Verwenden Sie für die Übersetzung die Methode `ICompilable`.

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

CompilerResult result = compileService.Compile();
```

Hinweis

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

Programmcode

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

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

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

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

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

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

Um das Übersetzungsergebnis auszuwerten, ändern Sie folgenden Programmcode:

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

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.10.12 Projekt speichern

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Ein Projekt speichern

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

Programmcode

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

```
public static void SaveProject(TiaPortal tiaPortal)
{
    Project project = 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();
    }
}
```

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

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

7.10.13 Projekt schließen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um ein Projekt zu schließen, ändern Sie folgenden Programmcode:

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

7.11 Funktionen für Verbindungen

7.11.1 Konfigurierbare Attribute einer Port-zu-Port-Verbindung

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Projekt öffnen (Seite 99)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

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

Einstellungen der Portschnittstelle

Die folgenden Attribute werden für die Einstellungen der Portschnittstelle bereitgestellt:

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

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

Wert	Beschreibung
`MediumAttachmentType.None`	Anbindungstyp kann nicht ermittelt werden.
`MediumAttachmentType.Copper`	Anbindungstyp ist Kupfer.
`MediumAttachmentType.FibreOptic`	Anbindungstyp ist Glasfaser.

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

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

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

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

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

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

Portoptionen

Die folgenden Attribute werden für Portoptionen bereitgestellt:

Attributname	Datentyp	Schreibbar	Zugriff
`PortActivation`	Bool	r/w	Dynamisches Attribut
`TransmissionRateAndDuplex`	TransmissionRateAndDuplex	r/w	Dynamisches Attribut

Attributname	Datentyp	Schreib-bar	Zugriff
`PortMonitoring`	Bool	r/w	Dynamisches Attribut
`TransmissionRateAutoNegotiation`	Bool	r/w	Dynamisches Attribut
`EndOfDetectionOfAccessibleDevices`	Bool	r/w	Dynamisches Attribut
`EndOfTopologyDiscovery`	Bool	r/w	Dynamisches Attribut
`EndOfSyncDomain`	Bool	r/w	Dynamisches Attribut

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

Wert	Beschreibung
`TransmissionRateAndDuplex.None`	
`TransmissionRateAndDuplex.Automatic`	Automatisch
`TransmissionRateAndDuplex.AUI10Mbps`	10 Mbps AUI
`TransmissionRateAndDuplex.TP10MbpsHalfDuplex`	TP 10 Mbps Halbduplex
`TransmissionRateAndDuplex.TP10MbpsFullDuplex`	TP 10 Mbps Vollduplex
`TransmissionRateAndDuplex.AsyncFiber10MbpsHalfDuplex`	Asynchrone Glasfaser 10 Mbit/s Halbduplexmodus
`TransmissionRateAndDuplex.AsyncFiber10MbpsFullDuplex`	Asynchrone Glasfaser 10 Mbit/s Vollduplexmodus
`TransmissionRateAndDuplex.TP100MbpsHalfDuplex`	TP 100 Mbps Halbduplex
`TransmissionRateAndDuplex.TP100MbpsFullDuplex`	TP 100 Mbps Vollduplex
`TransmissionRateAndDuplex.FO100MbpsFullDuplex`	FO 100 Mbps Vollduplex
`TransmissionRateAndDuplex.X1000MbpsFullDuplex`	X1000 Mbps Vollduplex
`TransmissionRateAndDuplex.FO1000MbpsFullDuplexLD`	FO 1000 Mbps Vollduplex LD
`TransmissionRateAndDuplex.FO1000MbpsFullDuplex`	FO 1000 Mbps Vollduplex
`TransmissionRateAndDuplex.TP1000MbpsFullDuplex`	TP 1000 Mbps Vollduplex
`TransmissionRateAndDuplex.FO10000MbpsFullDuplex`	FO 10000 Mbps Vollduplex
`TransmissionRateAndDuplex.FO100MbpsFullDuplexLD`	FO 100 Mbps Vollduplex LD
`TransmissionRateAndDuplex.POFPCF100MbpsFullDuplexLD`	POF/PCF 100 Mbps Vollduplex

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

7.12 Funktionen auf Bibliotheken

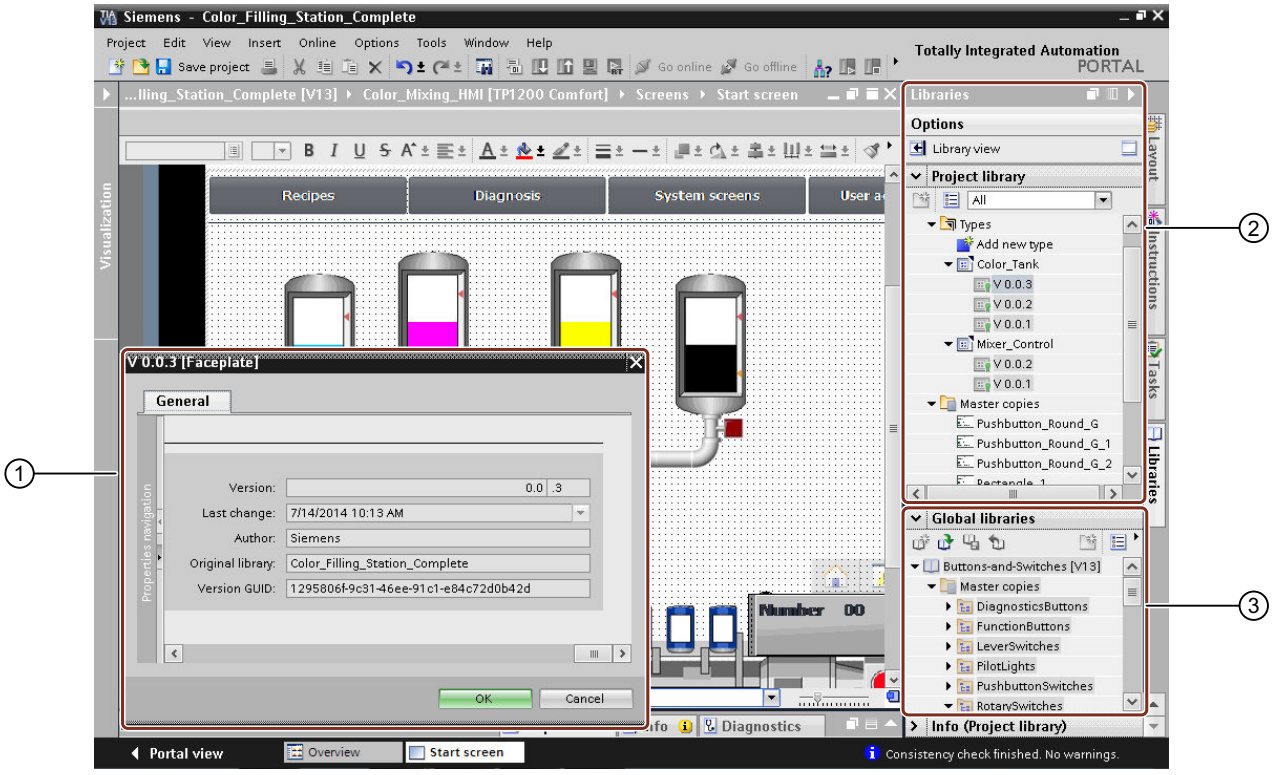
7.12.1 Funktionen auf Objekte und Instanzen

Zugriff auf Typen und Instanzen

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

Funktionen für Objekte und Instanzen

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



① Attribute von Typen, Typversionen, Masterkopien und Instanzen anzeigen

② Die folgenden Funktionen sind in der Projektbibliothek verfügbar:

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

③ Die folgenden Funktionen sind in der globalen Bibliothek verfügbar:

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

7.12.2 Auf globale Bibliotheken zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)

Verwendung

Es gibt drei Arten von globalen Bibliotheken.

- Globale Systembibliothek: Diese globalen Bibliotheken sind Bestandteil der Installation des TIA Portals und verwenden die Dateierweiterung *.as14. Alle globalen Systembibliotheken sind schreibgeschützt.
- Globale Unternehmensbibliothek: Diese globalen Bibliotheken wurden von einem Administrator ausgewählt und werden beim Starten des TIA Portals vorab geladen. Alle globalen Unternehmensbibliotheken sind schreibgeschützt.
- Globale Anwenderbibliothek: Diese globalen Bibliotheken wurden von Benutzern des TIA Portals erstellt. Globale Anwenderbibliotheken können entweder nur schreibgeschützt oder mit Lese- und Schreibrechten geöffnet werden.
  Wenn eine globale Anwenderbibliothek in einem bestimmten Modus bereits geöffnet ist, kann diese globale Anwenderbibliothek nicht in einem anderen Modus geöffnet werden.
  Globale Anwenderbibliotheken aus Vorgängerversionen können nur schreibgeschützt geöffnet werden.

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

Programmcode: Verfügbare globale Bibliotheken

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

```
TiaPortal tia = ...;
var availableLibraries = tia.GlobalLibraries.GetGlobalLibraryInfos();
foreach (GlobalLibraryInfo info in availableLibraries)
{
    //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);
}
```

Attribute von GlobalLibrary

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

Wert	Datentyp	Beschreibung
Name	String	Name der globalen Bibliothek.
Path	FileInfo	Pfad der globalen Bibliothek.

Attribute von GlobalLibraryInfo

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

Siehe auch

Auf Ordner in einer Bibliothek zugreifen (Seite 139)

7.12.3 Zugriff auf Sprachen der globalen Bibliothek

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Eine Bibliothek ist geöffnet.
  Siehe Bibliotheken öffnen (Seite 132)

Anwendung

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

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

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

- Zugriff auf eine Sammlung aktiver Sprachen.
- Suchen in der Sammlung aktiver Sprachen mithilfe von `System.Globalization.CultureInfo`.
- Hinzufügen einer Sprache zu einer Sammlung aktiver Sprachen.
- Entfernen einer Sprache aus einer Sammlung aktiver Sprachen.
- Festlegen einer Bearbeitungssprache.
- Festlegen einer Referenzsprache.

Attribut von Sprachen der globalen Bibliothek

Sprachen der globalen Bibliothek haben die folgenden Attribute:

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

Programmcode: Zugriff auf Spracheinstellungen

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

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

Programmcode: Sprachen der globalen Bibliothek enumerieren

Ändern Sie den folgende Programmcode, um die Sprachen der globalen Bibliothek zu enumerieren:

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

Programmcode: Festlegen der Sprachen der globalen Bibliothek

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

```
var globalLibrary = portal.GlobalLibraries.Open(m_GlobalLibrarypath, OpenMode.ReadOnly);
LanguageSettings languageSettings = globalLibrary.LanguageSettings;
LanguageComposition supportedLanguages = languageSettings.Languages;
LanguageAssociation activeLanguages = languageSettings.ActiveLanguages;
Language supportedGermanLanguage = supportedLanguages.Find(CultureInfo.GetCultureInfo("de-
DE"));
activeLanguages.Add(supportedGermanLanguage);
languagesSettings.EditingLanguage = supportedGermanLanguage;
languageSettings.ReferenceLanguage = supportedGermanLanguage;
```

Hinweis

Wenn Sie der globalen Bibliothek Sprachen hinzufügen oder sie darin ändern, bewirkt dies keine Änderungen an Sprachen von veröffentlichten Versionen der globalen Bibliothek. Dies gilt auch für die Aktualisierung von Sprachen der globalen Bibliothek über die Benutzeroberfläche (Spracheditor) oder den Navigator für die Spracheinstellungen (LanguageSettings).

7.12.4 Bibliotheken öffnen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet. Diese Voraussetzung gilt nur für den Zugriff auf Projektbibliotheken.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Eine globale Bibliothek kann über System.IO.FileInfo mit einem Pfad auf die Bibliotheksdatei auf einem lokalen Speichermedium oder einem Netzwerkspeicher geöffnet werden. Nur globale Anwenderbibliotheken können mittels Pfad geöffnet werden. Ein Pfad einer globalen Systembibliothek oder einer globalen Unternehmensbibliothek kann zum Öffnen der Bibliothek nicht verwendet werden.

Ab V14 SP1 können globale Bibliotheken mittels GlobalLibraryInfo geöffnet werden. Der OpenMode wird in der GlobalLibraryInfo angegeben.

Eine globale Anwenderbibliothek einer Vorgängerversion des TIA Portals kann hochgerüstet und mit der aktuellen Version des TIA Portals geöffnet werden. Eine globale Bibliothek der V13 oder einer Vorgängerversion kann nicht mittels Hochrüsten geöffnet werden. Diese Bibliotheken müssen zunächst auf V13 SP1 aktualisiert werden.

Mit TIA Portal Openness geöffnete globale Bibliotheken werden außerdem der Sammlung der globalen Bibliotheken im TIA Portal hinzugefügt und in der Benutzeroberfläche im TIA Portal angezeigt.

Programmcode: Bibliothek mittels System.IO.FileInfo öffnen

Ändern Sie folgenden Programmcode:

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

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

Programmcode: Bibliothek mittels GlobalLibraryInfo öffnen

Ändern Sie folgenden Programmcode:

```
TiaPortal tia = ...
IList<GlobalLibraryInfo> libraryInfos = tia.GlobalLibraries.GetGlobalLibraryInfos();
GlobalLibraryInfo libInfo = ...; //check for the info you need from the list, e.g.
GlobalLibrary libraryOpenedWithInfo;
if (libInfo.Name == "myLibrary")
libraryOpenedWithInfo = tia.GlobalLibraries.Open(libInfo);
```

Programmcode: Bibliothek hochrüsten

Ändern Sie folgenden Programmcode:

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

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

OpenMode

Wert	Beschreibung
ReadOnly	Schreibzugriff auf die Bibliothek.
ReadWrite	Lese- und Schreibzugriff auf die Bibliothek.

7.12.5 Offene Bibliotheken enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)

Anwendung

Alle geöffneten globalen Bibliotheken im TIA Portal, unabhängig davon, ob sie über API oder die Benutzeroberfläche geöffnet wurden, können enumeriert werden.

Globale Bibliotheken aus Vorgängerversionen des TIA Portals werden nicht enumeriert, wenn sie mit Schreibzugriff geöffnet wurden.

Programmcode

Ändern Sie den folgenden Programmcode, um offene globale Bibliotheken zu enumerieren:

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

Siehe auch

Projekt öffnen (Seite 99)

7.12.6 Bibliotheken speichern und schließen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Eine Bibliothek ist geöffnet.
  Siehe Bibliotheken öffnen (Seite 132)

Anwendung

Globale Anwenderbibliotheken können geschlossen oder gespeichert werden. Alle an der globalen Bibliothek vorgenommenen Änderungen werden nicht automatisch gespeichert. Alle nicht gespeicherten Änderungen werden ohne Benutzeraufforderung durch Schließen einer globalen Bibliothek verworfen.

Globale Systembibliotheken und globale Unternehmensbibliotheken können nicht geschlossen oder gespeichert werden.

Zum Speichern und Schließen einer globalen Bibliothek:

- Verwenden Sie die Methode Save ( ), wenn Sie eine globale Anwenderbibliothek speichern möchten
- Verwenden Sie die Methode SaveAS ( ), wenn Sie eine globale Anwenderbibliothek in einem anderen Verzeichnis speichern möchten
- Verwenden Sie die Methode Close ( ), um eine globale Anwenderbibliothek zu schließen

Programmcode

Ändern Sie folgenden Programmcode, um eine globale Anwenderbibliothek zu speichern:

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

Ändern Sie folgenden Programmcode, um eine globale Anwenderbibliothek an einem anderen Ort zu speichern:

```
TiaPortal portal = new TiaPortal(TiaPortalMode.WithUserInterface);
GlobalLibraryComposition globalLibraryComposition = portal.GlobalLibraries;
FileInfo existingLibraryfileInfo = new FileInfo(@"D:\GlobalLibraries\MyGlobalLibrary
\MyGlobalLibrary.al15");
DirectoryInfo targetDirectoryInfo = new DirectoryInfo(@"D:\GlobalLibraries
\GlobalLibrarySaveAs");
UserGlobalLibrary userGlobalLibary =
globalLibraryComposition.Open(existingLibraryfileInfo, OpenMode.ReadWrite);
userGlobalLibary.SaveAs(targetDirectoryInfo);
```

Ändern Sie folgenden Programmcode, um eine globale Anwenderbibliothek zu schließen:

```
UserGlobalLibrary userLib = ...
// close and discard changes
userLib.Close();
```

7.12.7 Bibliothek archivieren und abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Eine Bibliothek ist geöffnet.
  Siehe Bibliotheken öffnen (Seite 132)
- Eine Bibliothek ist gespeichert.
  Siehe Bibliotheken speichern und schließen (Seite 134)

Anwendung

Eine geöffnete und gespeicherte Bibliothek kann archiviert werden, bevor weitere Änderungen daran vorgenommen werden. Auf diese Weise verhindern Sie unbeabsichtigte Änderungen und können die archivierte Bibliothek später wieder abrufen. Außerdem können Sie die archivierte Datei problemlos im Netzwerk teilen.

Bibliothek archivieren

Über die Schnittstelle TIA Portal Openness API können Sie eine globale Anwenderbibliothek archivieren. Die API ist am Objekt "Siemens.Engineering.UserGlobalLibrary" verfügbar.

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

Hierbei ist 'targetName' der Name der Datei, die für die archivierten oder nicht-archivierten Bibliotheken erstellt wurde. Diese Datei kann Dateierweiterungen oder keine Dateierweiterungen enthalten. Wenn Sie keine Erweiterung oder eine andere Erweiterung als "zal15" oder "zal14" usw. angeben, kann die archivierte Datei nicht außerhalb der Openness API aus dem TIA Portal abgerufen werden.
Wenn der Wert für LibraryArchivationMode mit Compressed und DiscardRestorableDataAndCompressed angegeben wird, ist der Name der archivierten Datei identisch mit dem von Ihnen bereitgestellten Namen. Bei den Werten None und DiscardRestorableData für LibraryArchivationMode wird die Erweiterung der Bibliotheksdatei automatisch von der Komponente Project Manager des TIA Portals anhand der aktuellen Version des TIA Portals festgelegt.

Hinweis

Vor dem Aufrufen der Archive API müssen Sie die Bibliothek gespeichert haben. Falls die Bibliothek nicht gespeicherte Änderungen enthält, löst das Archiv eine EngineeringTargetInvocationException aus.

Archivierungsmodus der Bibliothek

Für die Enumeration LibraryArchivationMode (Archivierungsmodus der Bibliothek) können vier Werte angegeben werden.

LibraryArchivation-Mode	Beschreibung
Ohne	• Mit den Originaldateien werden keine speziellen Aktionen durchgeführt. Der Modus ähnelt dem Vorgang "Speichern unter". • In diesem Modus wird keine komprimierte ZIP-Datei erstellt. • Der Unterschied zu SaveAs besteht in diesem Fall darin, dass das Archiv den persistenten Speicherort nicht auf den neuen Ordner Archived umstellt, was jedoch bei SaveAs der Fall ist.
DiscardRestorable-Data	• Bei der Dateispeicherung wird die Bibliothek in einer internen Datendatei abgelegt. Diese Datei wird anschließend bei jeder Änderung an den Bibliotheksdaten größer. Beim Modus DiscardRestorableData wird diese Datendatei reorganisiert (dabei wird nur die neueste Version der Objekte gespeichert und der Verlauf aus der Datei gelöscht). Zwischenzeitlich angefallene Daten, die Dateien des IM-Verzeichnisses und des Verzeichnisses tmp (siehe Struktur der Bibliotheksverzeichnisse) werden nicht an den Archiv-Speicherort kopiert. • In diesem Modus wird keine komprimierte ZIP-Datei erstellt.
Compressed	Die vom Archivierungsprozess erstellte Struktur des TMP-Bibliotheksordners wird zu einem ZIP-kompatiblen Archiv komprimiert. Nach Erstellung der ZIP-Datei wird die TMP-Ordnerstruktur gelöscht.
DiscardRestorable-DataAndCompressed	Die vom Archivierungsprozess erstellte Struktur des TMP-Bibliotheksordners verwirft die wiederherstellbaren Daten und wird dann zu einem ZIP-kompatiblen Archiv komprimiert. Nach Erstellung der ZIP-Datei wird die TMP-Ordnerstruktur gelöscht.

Programmcode: Bibliothek archivieren

Ändern Sie folgenden Programmcode, um eine globale Anwenderbibliothek zu archivieren:

```
var tiaPortal = new TiaPortal(TiaPortalMode.WithoutUserInterface);
var libraryFilePath = @"E:\Sample1\Sample1.al15";
var userGlobalLibrary = tiaPortal.GlobalLibraries.Open(new FileInfo(LibraryFilePath),
OpenMode.ReadWrite);
var archivePath = @"E:\Archive";
var archiveFileName = "SampleArchive";
userGlobalLibrary.Archive(new DirectoryInfo(archivePath), archiveFileName,
LibraryArchivationMode.Compressed);
```

Bibliothek abrufen

Über die Schnittstelle TIA Portal Openness API können Sie eine archivierte TIA Portal-Bibliothek abrufen. Der Abruf ist nur mit einem komprimierten Archiv möglich. Die API ist am Objekt "Siemens.Engineering.GlobalLibraryComposition" verfügbar.

```
public Siemens.Engineering.Library.UserGlobalLibrary Retrieve(System.IO.FileInfo
sourcePath, System.IO.DirectoryInfo targetDirectory, Siemens.Engineering.OpenMode openMode)
```

Hinweis

Es ist nicht möglich, archivierte Bibliotheken mit den Enumerationswerten 'LibraryArchivationMode.None' oder 'LibraryArchivationMode.DiscardRestorableData' abzurufen.

Durch Aufrufen von RetrieveWithUpgrade API stellen Sie die archivierte Bibliothek einer früheren TIA Portal-Version wieder her. Die API-Definition lautet wie folgt:

```
public Siemens.Engineering.Library.UserGlobalLibrary
RetrieveWithUpgrade(System.IO.FileInfo sourcePath, System.IO.DirectoryInfo
targetDirectory, Siemens.Engineering.OpenMode openMode)
```

Öffnungsmodus

Die Enumeration OpenMode weist zwei Werte auf:

OpenMode	Beschreibung
ReadMode	• Lesezugriff auf die Bibliothek. Daten können aus der Bibliothek gelesen werden.
ReadWrite	• Schreibzugriff auf die Bibliothek. Daten können in die Bibliothek geschrieben werden.

Programmcode: Bibliothek abrufen

Ändern Sie den folgenden Programmcode, um eine Bibliothek abzurufen:

```
var archivePath = @"E:\Archive\Sample1.zal15";
var retrievedLibraryDirectory = @"E:\RetrievedLibraries";
var tiaPortal = new TiaPortal(TiaPortalMode.WithoutUserInterface);
tiaPortal.GlobalLibraries.Retrieve(new FileInfo(archivePath), new
DirectoryInfo(retrievedLibraryDirectory), OpenMode.ReadWrite));
```

Siehe auch

Bibliotheken öffnen (Seite 132)

Bibliotheken speichern und schließen (Seite 134)

7.12.8 Globale Bibliotheken erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)

Anwendung

Globale Bibliotheken können über die TIA Portal Openness API erzeugt werden, indem die Methode Create auf GlobalLibraryComposition aufgerufen wird. Es wird eine globale Anwenderbibliothek ausgegeben.

GlobalLibraryComposition.Create

Ändern Sie folgenden Programmcode:

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

Entsprechend diesem Beispiel

- wird ein Ordner "D:\GlobalLibraries\Library1" erstellt
- wird eine globale Bibliotheksdatei "D:\GlobalLibraries\Library1\Library1.alXX" erstellt

Parameter zum Erstellen globaler Bibliotheken

Parameter	Datentyp	Typ	Beschreibung
Author	String	Obligatorisch	Autor einer globalen Bibliothek.
Comment	String	Optional	Kommentar einer globalen Bibliothek.
Name	String	Optional	Name einer globalen Bibliothek.
TargetDirectory	DirectoryInfo	Obligatorisch	Verzeichnis, das den Ordner mit der globalen Bibliothek enthalten wird.

Siehe auch

Projekt öffnen (Seite 99)

7.12.9 Auf Ordner in einer Bibliothek zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128).

Verwendung

Über die Schnittstelle TIA Portal Openness API können Sie auf die Systemordner für Typen und Masterkopien in einer Bibliothek zugreifen. Dabei können Sie auf Typen, Typversionen, Masterkopien und benutzerdefinierte Ordner im Systemordner zugreifen.

Mit der Methode `Find`, z. B. `libTypeUserFolder.Folders.Find("SomeUserFolder");`, können Sie jederzeit auf einen benutzerdefinierten Ordner zugreifen.

Programmcode: Auf Systemordner zugreifen

Um auf die Systemordner für Typen in einer Bibliothek zuzugreifen, ändern Sie folgenden Programmcode:

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

Um auf die Systemordner für Masterkopien in einer Bibliothek zuzugreifen, ändern Sie folgenden Programmcode:

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

Programmcode: Über die Methode Find() auf benutzerdefinierte Ordner zugreifen

Ändern Sie folgenden Programmcode:

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

Programmcode: Benutzerdefinierte Ordner enumerieren

Um benutzerdefinierte Unterordner in einem Systemordner für Typen zu enumerieren, ändern Sie folgenden Programmcode:

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

Um benutzerdefinierte Unterordner in einem Systemordner für Masterkopien zu enumerieren, ändern Sie folgenden Programmcode:

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

Um benutzerdefinierte Unterordner in einem benutzerdefinierten Ordner für Typen zu enumerieren, ändern Sie folgenden Programmcode:

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

Um benutzerdefinierte Unterordner in einem benutzerdefinierten Ordner für Masterkopien zu enumerieren, ändern Sie folgenden Programmcode:

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

Programmcode: Benutzerdefinierte Ordner erstellen

Um einen benutzerdefinierten Ordner für Typen zu erstellen, ändern Sie folgenden Programmcode:

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

Um einen benutzerdefinierten Ordner für Masterkopien zu erstellen, ändern Sie folgenden Programmcode:

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

Programmcode: Benutzerdefinierte Ordner umbenennen

Um einen benutzerdefinierten Ordner für Typen zu erstellen, ändern Sie folgenden Programmcode:

```
var typeUserFolder =
project.ProjectLibrary.TypeFolder.Folders.Find("SampleTypeUserFolderName");
typeUserFolder.Name = "NewTypeUserFolderName";
```

```
var typeUserFolder = ProjectLibrary.TypeFolder.Folders.Find("SampleTypeUserFolderName");
typeUserFolder.SetAttributes(new[] {new KeyValuePair<string,object>("Name",
"NewTypeUserFolderName")});
```

Um einen benutzerdefinierten Ordner für Masterkopien zu erstellen, ändern Sie folgenden Programmcode:

```
var masterCopyUserFolder =
project.ProjectLibrary.MasterCopyFolder.Folders.Find("SampleMasterCopyUserFolderName");
masterCopyUserFolder.Name = "NewMasterCopyUserFolderName";
```

```
var masterCopyUserFolder =
ProjectLibrary.MasterCopyFolder.Folders.Find("SampleMasterCopyUserFolderName");
masterCopyUserFolder.SetAttributes(new[] {new KeyValuePair<string,object>("Name",
"NewMasterCopyUserFolderName")});
```

Siehe auch

Auf Kopiervorlagen zugreifen (Seite 152)

7.12.10 Auf Typen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128).
- Sie haben Zugriff auf eine Gruppe für Typen.
  Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 139).

Verwendung

Auf die Typen in einer Bibliothek können Sie über die Schnittstelle TIA Portal Openness API zugreifen.

- Sie können die Typen enumerieren.
- Sie können Typen umbenennen.
- Sie können auf die folgenden Attribute der einzelnen Typen zugreifen:

Attribut	Datentyp	Beschreibung
Author	String	Gibt den Namen des Autors zurück.
Comment	MultilingualText	Gibt den Kommentar zurück.
Guid	Guid	Gibt die GUID des Typs zurück.[1]
Name	String	Gibt den Namen des Typs zurück. [2]

[1] Mit diesem Attribut können Sie einen einzelnen Typ in einer Bibliothek finden. Die Suche ist rekursiv.

[2] Über dieses Attribut können Sie einen einzelnen Typ in einem Ordner finden. Unterordner werden nicht in die Suche einbezogen. Die Typennamen sind nicht eindeutig. Es kann mehrere Typen mit demselben Namen in verschiedenen Gruppen geben. Die GUID des Typs ist hingegen eindeutig.

Unterklassen für Bibliothekstypobjekte

Mit der TIA Portal Openness API können Sie über Unterklassen auf Bibliothekstypobjekte zugreifen. Folgende Unterklassen sind vorhanden:

- Siemens.Engineering.Hmi.Faceplate.FaceplateLibraryType
- Siemens.Engineering.Hmi.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

Der folgende Code ist ein Beispiel für die Verwendung von Unterklassen für Bibliothekstypen.

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

VBScriptLibraryType libraryTypeAsVbScript = libraryType as VBScriptLibraryType;
```

Programmcode

Um alle Typen im Systemordner einer Bibliothek zu enumerieren, ändern Sie folgenden Programmcode:

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

Um alle Typen in einem benutzerdefinierten Ordner einer Bibliothek zu enumerieren, ändern Sie folgenden Programmcode:

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

Um auf die Attribute eines Typs zuzugreifen, ändern Sie folgenden Programmcode:

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

Um einen einzelnen Typ anhand seines Namens oder der GUID zu finden, ändern Sie folgenden Programmcode:

```
public static void FindTypeObjectInLibrary(ILibrary library)
{
    // Find type object by its GUID in a given library:
    System.Guid targetGuid = ...;
    LibraryType libTypeByGUID = library.FindType(targetGuid);
    // Find type object by its name in a given group:
    LibraryTypeFolder libTypeSystemFolder = library.TypeFolder;
    LibraryType libTypeByName = libTypeSystemFolder.Types.Find("myTypeObject");
}
```

Um einen Typ umzubenennen, ändern Sie folgenden Programmcode:

```
// Setting the name attribute
var type = project.ProjectLibrary.TypeFolder.Types.Find("SampleTypeName");
type.Name = "NewTypeName";

//Setting the name attribute dynamically
var type = project.ProjectLibrary.TypeFolder.Types.Find("SampleTypeName");
type.SetAttributes(new[] {new KeyValuePair<string,object>("Name", "NewTypeName")});
```

7.12.11 Auf Typ-Versionen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128).
- Sie haben Zugriff auf eine Gruppe für Typen.
  Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 139).

Verwendung

Sie haben über die Schnittstelle TIA Portal Openness API Zugriff auf Typversionen.

- Sie können die Typversionen eines Typs enumerieren.
- Sie können den Typ ermitteln, zu dem eine Typversion gehört.
- Sie können die Instanzen einer Typversion enumerieren.
- Sie können eine neue Instanz einer Typversion erstellen.

- Sie können von einer Instanz zu dem damit verbundenen Versionsobjekt navigieren.
- Sie können auf die folgenden Attribute der einzelnen Typversionen zugreifen:

Attribut	Datentyp	Beschreibung
Author	String	Gibt den Namen des Autors zurück.
Comment	MultilingualText	Gibt den Kommentar zurück.
Guid	Guid	Gibt die GUID der Typversion zurück.[1]
ModifiedDate	DateTime	Gibt Datum und Uhrzeit zurück, zu denen die Typversion auf den Status "`Committed`" gesetzt wurde.
State	LibraryTypeVersionState	Gibt den Status der Version zurück: • `InWork`: Entspricht dem Status „In Bearbeitung" oder „Im Test“, abhängig von dem zugeordneten Typ. • `Committed`: Entspricht dem Status „Freigegeben".
TypeObject	LibraryType	Gibt den Typ zurück, zu dem diese Typversion gehört.
VersionNumber	Version	Gibt die Versionsnummer als dreistellige Versionskennung zurück, zum Beispiel „1.0.0".[2]

[1] Über dieses Attribut können Sie eine einzelne Typversion in einer Bibliothek finden.

[2] Über dieses Attribut können Sie eine einzelne Typversion in einer Zusammensetzung "LibraryTypeVersion" finden.

Alle Typversionen eines Typs enumerieren

Ändern Sie folgenden Programmcode:

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

Attribute einer Typversion aufrufen

Ändern Sie folgenden Programmcode:

```
//Acessing the attributes of a type version
public static void InspectPropertiesOfVersion(LibraryTypeVersion libTypeVersion)
{
    string versionAuthor = libTypeVersion.Author;
    MultilingualText versionComment = libTypeVersion.Comment;
    Guid versionGUID = libTypeVersion.Guid; DateTime versionModifiedDate =
libTypeVersion.ModifiedDate;
    LibraryTypeVersionState versionStateLibrary = libTypeVersion.State;
    LibraryType versionParentObject = libTypeVersion.TypeObject;
    Version versionNumber = libTypeVersion.VersionNumber;
}
```

Instanz einer Typversion erstellen

Sie können eine neue Instanz einer Typversion erstellen. Die folgenden Objekte werden unterstützt:

- Bausteine (FB/FC)
- PLC-Anwenderdatentypen
- Bilder
- VB-Skripte

Eine Instanz einer Typversion kann aus der globalen Bibliothek oder der Projektbibliothek erzeugt werden. Wenn Sie eine Instanz einer Typversion aus einer globalen Bibliothek erstellen, wird die Typversion zunächst mit der Projektbibliothek synchronisiert.

Eine wiederherstellbare Ausnahme wird ausgelöst, wenn eine Instanz nicht im Ziel erstellt werden kann. Mögliche Gründe sind:

- Die Bibliothekstypversion ist in Arbeit
- Eine Instanz der Bibliothekstypversion ist bereits im Zielgerät vorhanden

Ändern Sie folgenden Programmcode:

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

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

Ändern Sie folgenden Programmcode:

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

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

Ändern Sie folgenden Programmcode:

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

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

Ändern Sie folgenden Programmcode:

```
PlcTypeLibraryTypeVersion plcTypVersione=...;
PlcTypeComposition types=...;

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

Verwendungen einer Typversion ermitteln

Die folgenden Verwendungen werden bei Typversionen unterschieden:

- Die Typversion verwendet andere Typversionen aus der Bibliothek.
  Beispiel: Ein Anwenderdatentyp wird in einem Programmbaustein verwendet. Der Programmbaustein muss Zugriff auf den Anwenderdatentyp haben. Das bedeutet, dass der Programmbaustein vom Anwenderdatentyp abhängig ist.
  Wenn Sie mit der Methode `GetDependencies()` auf das Abhängigkeitsattribut von CodeBlockLibraryVersion zugreifen, wird eine Liste von LibraryTypeVersions zurückgegeben.
- Der Typ wird von einer anderen Typversion in der Bibliothek verwendet.
  Beispiel: Ein Anwenderdatentyp wird in einem Programmbaustein verwendet. Der Programmbaustein muss Zugriff auf den Anwenderdatentyp haben. Der Anwenderdatentyp hat den zugehörigen Programmbaustein. Der Programmbaustein ist vom Anwenderdatentyp abhängig.
  Wenn Sie mit der Methode `GetDependents()` auf das Abhängigkeitsattribut von PlcTypeLibraryTypeVersion zugreifen, wird eine Liste von LibraryTypeVersions zurückgegeben.

Bei beiden Attributen wird eine Liste zurückgegeben, die Objekte des Typs `LibraryTypeVersion` enthält. Sind keine Verwendungen vorhanden, wird eine leere Liste zurückgegeben.

Hinweis

Wenn Sie diese Attribute bei Typversionen mit dem Status "`InWork`" verwenden, kann eine Ausnahme ausgelöst werden.

Ändern Sie folgenden Programmcode:

```
//Determine the uses of a type version in a library
public static void GetDependenciesAndDependentsOfAVersion(LibraryTypeVersion
libTypeVersion)
{
    IList<LibraryTypeVersion> versionDependents = libTypeVersion.Dependents();
    IList<LibraryTypeVersion> versionDependencies = libTypeVersion.Dependencies();
}
```

Programmcode

Um den Typ zu ermitteln, zu dem eine Typversion gehört, ändern Sie folgenden Programmcode:

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

Um die Masterkopien zu ermitteln, die Instanzen einer Typversion enthalten, ändern Sie folgenden Programmcode:

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

Um eine einzelne Typversion anhand ihrer Versionsnummer zu finden, ändern Sie folgenden Programmcode:

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

7.12.12 Auf Instanzen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128).
- Sie haben Zugriff auf eine Gruppe für Typen.
  Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 139).

Verwendung

Sie haben über die Schnittstelle TIA Portal Openness API Zugriff auf Instanzen von Typversionen.

Mit der Methode `FindInstances(IInstanceSearchScope searchScope)` können Sie alle Instanzen einer Typversion ermitteln.

Mit dem Parameter `searchScope` können Sie den Bereich des Projekts angeben, der durchsucht werden soll. Die folgenden Klassen implementieren die Schnittstelle `IInstanceSearchScope` und können zum Suchen nach Instanzen verwendet werden:

- `PlcSoftware`
- `HmiTarget`

Die Methode gibt eine Liste mit Objekten des Typs `LibraryTypeInstanceInfo` zurück. Sind keine Instanzen vorhanden, wird eine leere Liste zurückgegeben.

Hinweis

Instanzen von Bildbausteinen und HMI-Benutzerdatentypen sind stets mit der zugehörigen Typversion verknüpft.

Instanzen aller anderen Objekte wie Programmbausteine oder Bilder können mit einer Typversion verknüpft sein.

Instanzen einer Typversion enumerieren

Ändern Sie folgenden Programmcode:

```
//Enumerate  the instances of a type version in the project
LibraryTypeVersion version = ...;
PlcSoftware plcSoftware = ...;

IInstanceSearchScope searchScope = plcSoftware as IInstanceSearchScope;

if(searchScope==null)
{
    //No search possible
}

IList<LibraryTypeInstanceInfo> instanceInfos = version.FindInstances(searchScope);
IEnumerable<IEngineeringObject> instances = instanceInfos.Select(instanceInfo =>
instanceInfo.LibraryTypeInstance);
```

Von einer Instanz zu dem damit verbundenen Versionsobjekt navigieren

Verwenden Sie das Attribut LibraryTypeVersion des Diensts LibraryTypeInstanceInfo, um von einer Instanz zu dem damit verbundenen Versionsobjekt zu navigieren.

Die folgenden Objekte bieten den Dienst LibraryTypeInstanceInfo:

- Bausteine FB
- Bausteine FC
- PLC-Anwenderdatentypen
- Bilder
- VB-Skripte

Wenn ein Instanzobjekt nicht mit einem Versionsobjekt verbunden ist, dann stellt es den Dienst "LibraryTypeInstanceInfo" nicht bereit.

```
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
}
```

Programmcode

Ändern Sie folgenden Programmcode in:

7.12.13 Auf Kopiervorlagen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128)
- Sie haben Zugriff auf eine Gruppe für Masterkopien.
  Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 139)

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt den Zugriff auf Masterkopien in einer globalen Bibliothek und der Projektbibliothek:

- Masterkopien erstellen
- Masterkopien in Systemordnern und benutzerdefinierten Ordnern enumerieren
- Masterkopien umbenennen
- Informationen von Masterkopien abfragen
- Informationen von Objekten in einer Masterkopie abfragen

Attribut	Datentyp	Beschreibung
Author	String	Gibt den Namen des Autors zurück.
ContentDescriptions	MasterCopyContentDescrip-tionComposition	Gibt eine Beschreibung des Inhalts der Masterkopie zurück.
CreationDate	DateTime	Gibt das Erstellungsdatum zurück.
Name	String	Gibt den Namen der Masterkopie zurück.

Programmcode

Um alle Masterkopien im Systemordner einer Bibliothek zu enumerieren, ändern Sie den folgenden Programmcode:

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

Ändern Sie den folgenden Programmcode, um mit der Methode "Find" auf eine einzelne Masterkopie zuzugreifen:

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

Ändern Sie den folgenden Programmcode, um Gruppen und Untergruppen von Masterkopien zu enumerieren:

```
private static void EnumerateFolder(MasterCopyFolder folder)
{
    EnumerateMasterCopies(folder.MasterCopies);
    foreach (MasterCopyUserFolder subFolder in folder.Folders)
    {
        EnumerateFolder(subFolder); // recursion
    }
}
private static void EnumerateMasterCopies(MasterCopyComposition masterCopies)
{
    foreach (MasterCopy masterCopy in masterCopies)
    {
        ...
    }
}
```

Ändern Sie den folgenden Programmcode, um mit der Methode "Find" auf einen MasterCopyUserFolder zuzugreifen:

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

Um eine Masterkopie umzubenennen, ändern Sie folgenden Programmcode:

```
//Setting the name attribute
var masterCopy = projectLibrary.MasterCopyFolder.MasterCopies.Find("SampleMasterCopyName");
masterCopy.Name = "NewMasterCopyName";

//Setting the name attribute dynamically
var masterCopy = projectLibrary.MasterCopyFolder.MasterCopies.Find("SampleMasterCopyName");
masterCopy.SetAttributes(new[] {new KeyValuePair<string,object>("Name",
"NewMasterCopyName")});
```

Informationen von Masterkopien abfragen

Um Informationen einer Masterkopie abzurufen, ändern Sie den folgenden Programmcode:

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

Informationen von Objekten in einer Masterkopie abfragen

Das Objekt MasterCopy enthält den Navigator ContentDescriptions, bei dem es sich um eine Zusammensetzung von MasterCopyContentDescriptions handelt.

Eine Masterkopie kann mehrere Objekte enthalten. Das Objekt MasterCopy enthält ContentDescriptions für jedes direkt in der Masterkopie enthaltene Objekt. Wenn die Masterkopie einen Ordner enthält, der ebenfalls einige Element enthält, enthält das Objekt MasterCopy nur eine ContentDescription des Ordners.

```
MasterCopy multiObjectMasterCopy = ...;

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

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

7.12.14 Masterkopie aus einem Projekt in Bibliothek erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Wenn eine Bibliothek eine Lesen-Schreiben-Bibliothek ist, können Sie eine Masterkopie aus einer IMasterCopySource-Quelle am Zielort erzeugen.

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

Eine EngineeringException-Ausnahme wird in folgenden Fällen ausgelöst:

- Am Zielort besteht nur Lesezugriff
- Die Erzeugung der Masterkopie aus der Quelle wird vom System abgelehnt

Die folgenden Elemente sind als IMasterCopySources definiert:

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

Programmcode

Verwenden Sie folgenden Programmcode:

```
// create a master copy from a code block in the project library
public static void Create(Project project, PlcSoftware plcSoftware)
{
    MasterCopySystemFolder masterCopyFolder = project.ProjectLibrary.MasterCopyFolder;
   CodeBlock block = plcSoftware.BlockGroup.Groups[0].Blocks.Find("Block_1") as CodeBlock;
    MasterCopy masterCopy = masterCopyFolder.MasterCopies.Create(block);
}
```

7.12.15 Ein Objekt aus einer Masterkopie erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt die Verwendung von Masterkopien im Projekt. Sie können in der Zusammensetzung des Objekts aus einer Masterkopie in einer Projektbibliothek oder einer globalen Bibliothek mit Hilfe der Methode CreateFrom ein Objekt erstellen.

Der Ausgabetyp entspricht dem Ausgabetyp der jeweiligen Zusammensetzung.

Die Methode CreateFrom unterstützt nur Masterkopien, die einzelne Objekte enthalten. Wenn die Zusammensetzung, wo die Aktion aufgerufen wird, und die Quellmasterkopie inkompatibel sind (z. B. wenn die Quellmasterkopie eine PLC-Variablentabelle enthält und die Zusammensatzung die Zusammensetzung eines PLC-Bausteins ist), wird eine wiederherstellbare Ausnahme ausgelöst.

Die folgenden Zusammensetzungen werden unterstützt:

- Siemens.Engineering.HW.DeviceComposition
- Siemens.Engineering.HW.DeviceItemComposition
- Siemens.Engineering.SW.Blocks.PlcBlockComposition
- Siemens.Engineering.SW.Tags.PlcTagTableComposition
- Siemens.Engineering.SW.Tags.PlcTagComposition
- Siemens.Engineering.SW.Types.PlcTypeComposition
- Siemens.Engineering.SW.TechnologicalObjects.TechnologicalInstanceDBComposition

- Siemens.Engineering.SW.Tags.PlcUserConstantComposition
- Siemens.Engineering.Hmi.Tag.TagTableComposition
- Siemens.Engineering.Hmi.Tag.TagComposition
- Siemens.Engineering.Hmi.Screen.ScreenComposition
- Siemens.Engineering.Hmi.Screen.ScreenTemplateComposition
- Siemens.Engineering.Hmi.RuntimeScripting.VBScriptComposition
- Siemens.Engineering.HW.SubnetComposition
- Siemens.Engineering.HW.DeviceUserGroupComposition
- Siemens.Engineering.SW.Blocks.PlcBlockUserGroupComposition
- Siemens.Engineering.SW.ExternalSources.PlcExternalSourceUserGroupComposition
- Siemens.Engineering.SW.Tags.PlcTagTableUserGroupComposition
- Siemens.Engineering.SW.Types.PlcTypeUserGroupComposition

Programmcode: PLC-Baustein aus einer Masterkopie erstellen

Um einen PLC-Baustein aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie den folgenden Programmcode:

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

Programmcode: Ein Gerät aus einer Masterkopie erstellen

Um ein Gerät aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie den folgenden Programmcode:

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

Programmcode: Ein Geräteelement aus einer Masterkopie erstellen

Um ein Geräteelement aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie den folgenden Programmcode:

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

Programmcode: Ein Subnetz aus einer Masterkopie erstellen

Um ein Subnetz aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie den folgenden Programmcode:

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

Programmcode: Einen Geräteordner aus einer Masterkopie erstellen

Um einen Geräteordner aus einer Masterkopie in einer Bibliothek zu erstellen, ändern Sie den folgenden Programmcode:

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

Siehe auch

Auf Kopiervorlagen zugreifen (Seite 152)

7.12.16 Masterkopien kopieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt das Kopieren von Masterkopien innerhalb einer Bibliothek und von einer Bibliothek zu einer anderen mithilfe der Aktion CreateFrom. Die Aktion erstellt ein neues Objekt basierend auf der Quellmasterkopie und platziert es in der Zusammensetzung, wo die Aktion aufgerufen wurde. Die Aktion versucht, die neue Masterkopie mit dem gleichen Namen zu erstellen wie die Quellmasterkopie. Wenn dieser Name nicht verfügbar ist, gibt das System der neuen Masterkopie einen neuen Namen. Danach gibt es die neue Masterkopie aus.

Wenn sich die Zusammensetzung, in der die Aktion "CreateFrom" aufgerufen wird, in einer schreibgeschützten globalen Bibliothek befindet, wird eine wiederherstellbare Ausnahme ausgelöst.

Programmcode

Ändern Sie folgenden Programmcode:

```
ProjectLibrary projectLibrary = ...;

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

Siehe auch

Auf Kopiervorlagen zugreifen (Seite 152)

7.12.17 Ermitteln veralteter Typinstanzen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128)
- Sie haben Zugriff auf einen Ordner für Typen.
  Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 139).

Verwendung

Über die Schnittstelle TIA Portal OpennessAPI können Sie Typversionen ermitteln, die zu den Instanzen im geöffneten Projekt gehören. Dabei liefert die TIA Portal OpennessAPI pro Instanz einen der beiden folgenden Zustände:

- Die Instanz bezieht sich auf eine veraltete Typversion.
- Die Instanz bezieht sich auf die aktuelle Typversion.

Für die Versionsermittlung gelten die folgenden Regeln:

- Basis für die Versionsermittlung sind eine Bibliothek und das Projekt, das Sie über die Schnittstelle TIA Portal OpennessAPI öffnen möchten.
- Im Rahmen der Versionsermittlung werden keine Instanzen aktualisiert.

Signatur

Über die Methode `UpdateCheck` können Sie die Instanzen einer Typversion ermitteln:
`UpdateCheck(Project project, UpdateCheckMode updateCheckMode)`

Parameter	Funktion
`Project`	Gibt das Projekt an, in dem die Typversionen von Instanzen ermittelt werden.
`UpdateCheckMode`	Gibt die Versionen an, die ermittelt werden: • `ReportOutOfDateOnly`: Gibt nur den Status vom Typ „veraltet" zurück. • `ReportOutOfDateAndUpToDate`: Gibt den Status der Typen „veraltet" und „aktuell“ zurück.

Ergebnis

Bei der Versionsermittlung werden die Geräte des Projekts von oben nach unten abgefragt. Jedes Gerät wird darauf geprüft, ob seine Konfigurationsdaten eine Instanz einer Typversion aus der angegebenen Bibliothek enthalten. Die Methode `UpdateCheck` gibt das Ergebnis der Versionsprüfung in hierarchischer Reihenfolge zurück.

Die nachstehende Tabelle zeigt das Ergebnis einer Versionsprüfung mit dem Parameter `UpdateCheck.ReportOutOfDateAndUpToDate`:

Update check for: HMI_1				
	Update check for library element Screen_1 0.0.3			
		Out-of-date		
			\HMI_1\Screens	Screen_4 0.0.1
			\HMI_1\Screens	Screen_2 0.0.2
		Up-to-date		
			\HMI_1\Screens	Screen_1 0.0.3
			\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

Programmcode

Es besteht kein Unterschied zwischen Projektbibliotheken und globalen Bibliotheken hinsichtlich der Handhabung der Aktualisierungsprüfung.

Um die Typversionen einer globalen Bibliothek oder Projektbibliothek für Instanzen im Projekt zu ermitteln, ändern Sie den folgenden Programmcode:

```
public static void UpdateCheckOfGlobalLibrary(Project project, ILibrary library)
{
    // check for out of date instances and report only out of date instances in the returned
feedback
    UpdateCheckResult result = library.UpdateCheck(project,
UpdateCheckMode.ReportOutOfDateOnly);

    //Alternatively, check for out of date instances and report both out of date and up to
date instances in the returned feedback
    UpdateCheckResult alternateResult = library.UpdateCheck(project,
UpdateCheckMode.ReportOutOfDateAndUpToDate);

    //Show result
    RecursivelyWriteMessages(result.Messages);

    // Alternatively, show result and access single message parts
    RecursivelyWriteMessageParts(result.Messages);
}
```

Um das Ergebnis der Versionsermittlung auszugeben und die Meldungen einzeln zu durchlaufen, ändern Sie folgenden Programmcode:

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

Um im Ergebnis der Versionsermittlung auf einzelne Meldungsteile zuzugreifen, ändern Sie folgenden Programmcode:

```
private static void RecursivelyWriteMessageParts (UpdateCheckResultMessageComposition
messages, string indent= "")
{
    indent += "\t";
    foreach (UpdateCheckResultMessage message in messages)
    {
        Console.WriteLine(indent + "Full description: " + message.Description);
        foreach (KeyValuePair<string, string> messagePart in message.MessageParts)
        {
             // first level
             // part 1: device name
             // second level:
             // part 1: Name of the type in the global library
             // part 2: version of the type in the global library
             // third level:
             // part 1: title (either "Out-of-date" or "Up-to-date");
             // fourth level:
             // part 1: Path hierarchy to instance
             // part 2: Instance name in project
             // part 3: Version of the instance in the project
             Console.WriteLine(indent + "*Key: {0} Value:{1}", messagePart.Key,
messagePart.Value);
        }
     RecursivelyWriteMessageParts(message.Messages,indent);
    }
}
```

7.12.18 Projekt aktualisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128).
- Sie haben Zugriff auf einen Ordner für Typen.
  Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 139).

Verwendung

Über die Schnittstelle TIA Portal Openness API können Sie Instanzen eines ausgewählten Typs in einem Typordner eines Projekts aktualisieren.

Bei der Aktualisierung werden die im Projekt verwendeten Instanzen basierend auf der letzten freigegebenen Typversion aktualisiert. Wenn Sie die Instanzen in einer globalen Bibliothek aktualisieren, wird zunächst eine Synchronisierung durchgeführt.

Signatur

Zum Aktualisieren von Instanzen verwenden Sie die Methode `UpdateProject`.

Verwenden Sie für Klassen, die die Schnittstelle `LibraryTypes` implementieren, den folgenden Aufruf:

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

Verwenden Sie für Klassen, die die Schnittstelle `ILibrary` implementieren, den folgenden Aufruf:

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

Jeder Aufruf wird in die Protokolldatei im Projektverzeichnis eingetragen.

Parameter	Funktion
`IEnumerable<ILibraryTypeOrFolderSelection> selectedTypesOrFolders`	Gibt den Ordner oder zu synchronisierende Typen oder deren Instanzen im Projekt an, die aktualisiert werden sollen.
`IUpdateProjectScope updateProjectScope` `IEnumerable <IUpdateProjectScope> updateProjectScope`	Gibt die Objekte im Projekt an, in denen die Verwendungen der Instanzen aktualisiert werden sollen. Die folgenden Objekte werden unterstützt: • `PlcSoftware` • `HmiTarget`

Programmcode

Um Instanzen von ausgewählten Typen innerhalb eines Typordners zu aktualisieren, ändern Sie folgenden Programmcode:

```
private static void UpdateInstances(ILibrary myLibrary, LibraryTypeFolder
singleFolderContainingTypes, LibraryType singleType, PlcSoftware plcSoftware, HmiTarget
hmiTarget)
{
    //Update Instances of multiple types (subset of types and folders)
    IUpdateProjectScope[] updateProjectScopes =
    {
        plcSoftware as IUpdateProjectScope, hmiTarget as IUpdateProjectScope
    };
    myLibrary.UpdateProject(new ILibraryTypeOrFolderSelection[] {singleType,
singleFolderContainingTypes}, updateProjectScopes);
    //Update Instances of multiple types (all types in library)
    myLibrary.UpdateProject(new[] {myLibrary.TypeFolder}, updateProjectScopes);
}
```

7.12.19 Bibliothek aktualisieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128).
- Sie haben Zugriff auf einen Ordner für Typen.
  Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 139).

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt die folgenden Aktualisierungen in der Projektbibliothek:

- Synchronisieren ausgewählter Typen zwischen Bibliotheken.

Bei der Synchronisierung wird die Ordnerstruktur nicht angepasst. Die zu aktualisierenden Typen werden über ihre GUID ermittelt und aktualisiert:

- Wenn ein Typ in einer Bibliothek eine Typversion enthält, die in der zu aktualisierenden Bibliothek fehlt, wird die Typversion kopiert.
- Der Vorgang wird abgebrochen und eine Exception ausgelöst, wenn ein Typ in einer Bibliothek eine Typversion mit unterschiedlicher GUID enthält.

Signatur

Zum Synchronisieren von Typversionen verwenden Sie die Methode `UpdateLibrary`.

Verwenden Sie für Klassen, die die Schnittstelle `LibraryTypes` implementieren, den folgenden Aufruf:

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

Verwenden Sie für Klassen, die die Schnittstelle `ILibrary` implementieren, den folgenden Aufruf:

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

Parameter	Funktion
`IEnumerable<ILibraryTypeOrFolderSelection> selectedTypesOrFolders`	Gibt den Ordner oder zu synchronisierende Typen oder deren Instanzen im Projekt an, die aktualisiert werden sollen.
`ILibrary targetLibrary`	Gibt die Bibliothek an, deren Inhalt mit einer Bibliothek synchronisiert wird. Wenn Quellbibliothek und Zielbibliothek identisch sind, wird eine Ausnahme ausgelöst.

Programmcode

Um einen Typ aus einer Projektbibliothek mit einer globalen Bibliothek zu synchronisieren, ändern Sie den folgenden Programmcode:

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

Um ausgewählte Typen in einem Typordner zwischen einer globalen Bibliothek und der Projektbibliothek zu synchronisieren, ändern Sie folgenden Programmcode:

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

7.12.20 Bibliotheksinhalte löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Sie haben Zugriff auf die erforderliche Bibliothek.
  Siehe Auf globale Bibliotheken zugreifen (Seite 128).
- Sie haben Zugriff auf einen Ordner für Typen.
  Siehe Auf Ordner in einer Bibliothek zugreifen (Seite 139).

Verwendung

Mit der Schnittstelle TIA Portal Openness API können Sie die folgenden Inhalte in der Projektbibliothek löschen:

- Typen
- Typversion
- Benutzerdefinierte Ordner für Typen
- Masterkopien
- Benutzerdefinierte Ordner für Masterkopien

Hinweis

Löschen von Typen und Ordnern mit benutzerdefinierten Typen

Wenn Sie einen Typ oder einen Ordner mit benutzerdefinierten Typen löschen möchten, müssen die "Regeln zum Löschen von Versionen" erfüllt werden. Einen leeren Typordner können Sie jederzeit löschen.

Hinweis

Regeln zum Löschen von Versionen

Sie können nur Versionen mit dem Status „Committed" löschen. Die folgenden Regeln gelten ebenfalls beim Löschen von Versionen:

- Wenn eine neue Version mit dem Status "`InWork`" gerade aus einer Version mit dem Status "`Committed`" erstellt wurde, können Sie die Version mit dem Status "`Committed`" nur dann löschen, wenn die neue Version verworfen wird oder den Status "`Committed`" erhält.
- Wenn ein Typ lediglich eine Version hat, wird auch der Typ gelöscht.
- Wenn Version A von Version B eines anderen Typs abhängig ist, löschen Sie zunächst Version A und danach Version B.
- Wenn es Instanzen von Version A gibt, können Sie Version A nur dann löschen, wenn auch die Instanzen gelöscht werden. Wenn eine Instanz auch in einer Masterkopie enthalten ist, wird die Masterkopie ebenfalls gelöscht.

Programmcode

Ändern Sie den folgenden Programmcode, um Typen oder Ordner mit benutzerdefinierten Typen zu löschen:

```
public static void DeleteMultipleTypesOrTypeUserFolders(ILibrary library)
{
    LibraryType t1 = library.TypeFolder.Types.Find("type1");
    LibraryType t2 = library.TypeFolder.Types.Find("type2");
    LibraryTypeUserFolder f1 = library.TypeFolder.Folders.Find("folder1");
    t1.Delete();
    t2.Delete();
    f1.Delete();
}
```

Um einen einzelnen Typ oder einen einzelnen Ordner mit benutzerdefinierten Typen zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteSingleTypeOrTypeUserFolder(ILibrary library)
{
    //Delete a single type
    LibraryType t1 = library.TypeFolder.Types.Find("type1");
    t1.Delete();

    //Delete a single folder
    LibraryTypeFolder parentFolder = library.TypeFolder;
    LibraryTypeUserFolder f1 = parentFolder.Folders.Find("folder1");
    f1.Delete();
}
```

Um eine Version zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteVersion(ILibrary library)
{
    LibraryType singleType = library.TypeFolder.Types.Find("type1");
    LibraryTypeVersion version1 = singleType.Versions.Find(new System.Version(1, 0, 0));
    version1.Delete();
}
```

Um eine Masterkopie oder einen Ordner mit benutzerdefinierten Masterkopien zu löschen, ändern Sie den folgenden Programmcode:

```
public static void DeleteMasterCopies(ILibrary library)
{
    // Delete master copy
    MasterCopy masterCopy = library.MasterCopyFolder.MasterCopies.Find("myMasterCopy");
    masterCopy.Delete();

    // Delete master copy user folder
    MasterCopyUserFolder masterUserFolder =
library.MasterCopyFolder.Folders.Find("myFolder");
    masterUserFolder.Delete();
}
```

Siehe auch

Auf Kopiervorlagen zugreifen (Seite 152)

7.13 Funktionen für das Aufrufen von Geräten, Netzwerken und Verbindungen

7.13.1 Editor "Geräte & Netze" öffnen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Sie können über die API-Schnittstelle den Editor „Geräte & Netze“ über eine der beiden folgenden Methoden öffnen:

- `ShowHwEditor(View.Topology` oder `View.Network` oder `View.Device)`: Öffnet den Editor „Geräte & Netze“ aus dem Projekt.
- `ShowInEditor(View.Topology` oder `View.Network` oder `View.Device)`: Zeigt das angegebene Gerät im Editor „Geräte & Netze“ an.

Über den Parameter `View` definieren Sie die Sicht, die beim Öffnen des Editors angezeigt wird:

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

Programmcode

Um den Editor „Geräte & Netze“ zu öffnen, ändern Sie den folgenden Programmcode:

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

Um den Editor „Geräte & Netze“ für ein Gerät zu öffnen, ändern Sie folgenden Programmcode:

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

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.13.2 PLC-Target und HMI-Target abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Sie können ermitteln, ob eine Softwarebasis in der TIA Portal Openness API als PLC-Target (PlcSoftware) oder HMI-Target verwendet werden kann.

Programmcode: PLC-Target

Um festzustellen, ob ein Gerätelement als PLC-Target verwendet werden kann, verwenden Sie folgenden Programmcode:

```
// Returns PlcSoftware
private PlcSoftware GetPlcSoftware(Device device)
{
    DeviceItemComposition deviceItemComposition = device.DeviceItems;
    foreach (DeviceItem deviceItem in deviceItemComposition)
    {
        SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
        if (softwareContainer != null)
        {
            Software softwareBase = softwareContainer.Software;
            PlcSoftware plcSoftware = softwareBase as PlcSoftware;
            return plcSoftware;
        }
    }
    return null;
 }
```

Programmcode: HMI-Target

Um festzustellen, ob ein Geräteelement als HMI-Target verwendet werden kann, ändern Sie folgenden Programmcode:

```
//Checks whether a device is of type hmitarget
private HmiTarget GetHmiTarget(Device device)
{
    DeviceItemComposition deviceItemComposition = device.DeviceItems;
    foreach (DeviceItem deviceItem in deviceItemComposition)
    {
        SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
        if (softwareContainer != null)
        {
            Software softwareBase = softwareContainer.Software;
            HmiTarget hmiTarget = softwareBase as HmiTarget;
            return hmiTarget;
        }
    }
    return null;
}
```

Siehe auch

Geräte enumerieren (Seite 220)

7.13.3 Auf Attribute eines Adressobjekts zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Für Schreibzugriff ist der PLC offline.

Verwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Attribute des Adressobjekts abrufen oder festlegen.

Ferner können Sie das aktuelle Prozessabbild einem OB zuweisen.

Auf die folgenden Attribute kann zugegriffen werden:

Attributname	Datentyp	Schreibbar	Zugriff	Beschreibung
`IsochronousMode`	BOOL	r/w	Dynamisches Attribut	Isochronen Modus aktivieren/ deaktivieren
`ProcessImage`	Int32	r/w	Dynamisches Attribut	Partitionsnummer des Prozessabbilds festlegen/abrufen
`InterruptObNumber`	Int64	r/w	Dynamisches Attribut	Nummer des Alarm-Organisationsbausteins festlegen/abrufen (nur klassischer Controller)
`StartAddress`	Int32	r/w	Modelliertes Attribut	Neuen Wert für StartAddress festlegen/abrufen

Einschränkungen

- Attribut `StartAddress`
  - Durch Festlegen von `StartAddress` kann implizit die `StartAddress` des gegenteiligen IO-Typs am Namensmodul geändert werden. Durch Ändern der Eingangsadresse wird die Ausgangsadresse geändert.
  - Der Schreibzugriff wird nicht für alle Geräte unterstützt.
  - Gepackte Adressen werden in TIA Portal Openness nicht unterstützt.
  - Durch Ändern der Adresse über TIA Portal Openness werden die zugewiesenen Variablen nicht neu verdrahtet.
- Attribut `InterruptObNumber`
  - Nur zugänglich in Einstellungen mit S7-300 oder S7-400 Controllern. Bei S7-400 Controllern wird der Schreibzugriff unterstützt.

Programmcode: Attribute eines Adressobjekts abrufen oder festlegen

Um auf den isochronen Modus eines Adressobjekts zuzugreifen, ändern Sie den folgenden Programmcode:

```
Address address= ...;

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

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

Um auf das Attribut `ProcessImage` eines Adressobjekts zuzugreifen, ändern Sie den folgenden Programmcode:

```
Address address= ...;

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

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

Um auf das Attribut `InterruptObNumber` eines Adressobjekts zuzugreifen, ändern Sie den folgenden Programmcode:

```
Address address= ...;

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

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

//default value = 40
```

Um auf das Attribut `StartAddress` eines Adressobjekts zuzugreifen, ändern Sie den folgenden Programmcode:

```
Address address= ...;

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

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

Programmcode: Das aktuelle Prozessabbild einem OB zuweisen

Um das aktuelle Prozessabbild einem OB zuweisen, ändern Sie den folgenden Programmcode:

```
OB obX =…
Address address= ...;

// assign PIP 5 to obX

address.SetAttribute("ProcessImage", 5);

try
{
        address.AssignProcessImageToOrganizationBlock(obX);
}    catch(RecoverableException e) {
       Console.WriteLine(e.Message);
}

// remove this PIP-OB assignment

try
{
        address.AssignProcessImageToOrganizationBlock(null);
}    catch(RecoverableException e) {
       Console.WriteLine(e.Message);
}
```

7.13.4 Aufrufen eines Modulkanals

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Signalmodule wie analoge Eingangsmodule haben normalerweise mehrere Kanäle in einem einzelnen Modul. Normalerweise bieten Kanäle mehrere Funktionen gleichzeitig, z. B. kann ein analoges Eingangsmodul mit vier Kanälen vier Spannungswerte zur selben Zeit messen.

Zum Aufrufen aller Kanäle eines Moduls wird das Kanalattribut eines Geräteelements verwendet.

Programmcode: Attribute von Kanälen

Um auf Attribute eines Kanals zuzugreifen, ändern Sie folgenden Programmcode:

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

Programmcode: Attribute ermitteln

Um die identifizierenden Attribute für die einzelnen Kanäle abzurufen, ändern Sie folgenden Programmcode:

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

Programmcode: Aufrufen eines einzelnen Kanals

Um die identifizierenden Attribute für den direkten Zugriff auf einen Kanal zu verwenden, ändern Sie folgenden Programmcode:

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

Kanaltypen

Wert	Beschreibung
ChannelType.None	Der Kanaltyp ist ungültig.
ChannelType.Analog	Der Kanaltyp ist analog.
ChannelType.Digital	Der Kanaltyp ist digital.
ChannelType.Technology	Der Kanaltyp ist technologisch.

Kanal-IO-Typen

Wert	Beschreibung
ChannelIOType.None	Der Kanal-IO-Typ ist ungültig.
ChannelIOType.Input	Ein Eingangskanal.
ChannelIOType.Output	Ein Ausgangskanal.
ChannelIOType.Complex	Komplexe IO-Typen, z. B. für technologische Kanäle.

7.13.5 Arbeiten mit Assoziationen

Auf Zuordnungen zugreifen

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

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

Attribute

Die folgenden Attribute sind verfügbar:

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

Methoden

TIA Portal Openness unterstützt die folgenden Methoden:

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

[1]: Wird nicht von allen Zuordnungen unterstützt.

7.13.6 Mit Zusammensetzungen arbeiten

Zusammensetzungen aufrufen

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

Attribute

Die folgenden Attribute sind verfügbar:

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

Methoden

TIA Portal Openness unterstützt die folgenden Methoden:

- `retType Create (id, …)`: Erzeugt eine neue Instanz und fügt diese Instanz der Zusammensetzung hinzu.
  Die Signatur der Methode ist abhängig von der Art und Weise, in der die Instanz erzeugt wird. Diese Methode wird nicht von allen Zusammensetzungen unterstützt.
- `type Find (id, …)`: Durchsucht eine Zusammensetzung nach der Instanz mit der übertragenen ID.
  Die Suche ist nicht rekursiv. Die Signatur der Methode ist abhängig von der Weise, in der nach der Instanz gesucht wird. Diese Methode wird nicht von allen Zusammensetzungen unterstützt.
- `IEnumerator GetEnumerator<retType> ()`: Kommt in foreach-Schleifen zum Einsatz und ermöglicht den Zugriff auf ein Objekt.
- `Delete (type)`[1]: Löscht die von der aktuellen Objektreferenz spezifizierte Instanz.
- `int IndexOf (type)`: Gibt den Index in der Zusammensetzung für eine übertragene Instanz zurück.
- `bool Contains (type)`: Ermittelt, ob die übertragene Instanz in der Zusammensetzung enthalten ist.
- `void Import(string path, ImportOptions importOptions)`[1]: Wird für jede Zusammensetzung verwendet, die importierbare Typen enthält.
  Jede Importsignatur enthält einen Konfigurationsparameter des Typs "ImportOptions (Seite 429)" ("None", "Overwrite"), mit dem der Benutzer das Importverhalten steuert.

[1]: Nicht von allen Zusammensetzungen unterstützt.

7.13.7 Objektgleichheit prüfen

Verwendung

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

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

Programmcode

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

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

7.13.8 Lese-Operationen für Attribute

Gruppenoperationen und Standardleseoperationen für Attribute

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

- Gruppenoperation für Lesezugriff
- Standardleseoperationen

Programmcode für Gruppenoperationen

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

Gruppenoperation für Lesezugriff

Diese Methode ist für jedes Objekt verfügbar:

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

Standardleseoperationen

Die folgenden Operationen sind verfügbar:

- Namen der verfügbaren Attribute abrufen:
  Verwenden Sie die Methode GetAttributeInfos() (Seite 96) auf einem `IEngineeringObject`.
- Generische Methode zum Lesen eines Attributs
  `public abstract object GetAttribute(string name);`

Hinweis

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

7.14 Funktionen in Netzwerken

7.14.1 Subnetz anlegen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Subnetze können auf zwei verschiedene Arten angelegt werden:

- Erstellen eines Subnetzes, das mit einer Schnittstelle verbunden ist: Die Art der Schnittstelle, an der das Subnetz angelegt wird, bestimmt die Art des Subnetzes
- Erstellen eines Subnetzes, das nicht mit einer Schnittstelle verbunden ist.

Programmcode: Erstellen eines Subnetzes, das mit einer Schnittstelle verbunden ist

Um ein Subnetz zu erstellen, ändern Sie folgenden Programmcode:

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

Die folgenden Typkennungen werden verwendet:

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

Programmcode: Erstellen eines Subnetzes, das nicht mit einer Schnittstelle verbunden ist

Um ein Subnetz zu erstellen, ändern Sie folgenden Programmcode:

```
Project project = ...;
SubnetComposition subnets = project.Subnets;

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

Die folgenden Typkennungen können verwendet werden:

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

7.14.2 Auf Subnetze zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Für mehrere netzwerkbezogene Merkmale, z. B. Zuordnung von Schnittstellen zu einem Subnetz, müssen Sie Subnetze im Projekt aufrufen. Subnetze sind typischerweise direkt auf Projektebene zusammengefasst.

Programmcode: Auf alle Subnetze eines Projekts zugreifen

Um auf alle Subnetze mit Ausnahme von internen Subnetzen eines Projekts zuzugreifen, ändern Sie folgenden Programmcode:

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

Programmcode: Auf ein spezifisches Subnetz zugreifen

Um auf ein bestimmtes Subnetz anhand des Namens zuzugreifen, ändern Sie folgenden Programmcode:

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

Attribute eines Subnetzes

Ein Subnetz hat folgende Attribute:

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

Netzwerktypen

Wert	Beschreibung
NetType.Unknown	Der Typ des Netzwerks ist unbekannt.
NetType.Ethernet	Der Typ des Netzwerks ist Ethernet.
NetType.Profibus	Der Typ des Netzwerks ist Profibus.
NetType.Mpi	Der Typ des Netzwerks ist MPI.
NetType.ProfibusIntegrated	Der Typ des Netzwerks ist integrierter Profibus.
NetType.Asi	Der Typ des Netzwerks ist ASi.
NetType.PcInternal	Der Typ des Netzwerks ist PC intern.
NetType.Ptp	Der Typ des Netzwerks ist PtP.
NetType.Link	Der Typ des Netzwerks ist Link.
NetType.Wan	Der Typ des Netzwerks ist Wide Area Network.

7.14.3 Auf interne Subnetze zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Wenn ein Geräteelement ein Subnetz bilden kann, verfügt es über die zusätzliche Funktionalität "Subnetzeigentümer". Um auf diese zusätzliche Funktionalität zuzugreifen, muss ein bestimmter Dienst des Geräteelements verwendet werden.

Programmcode: Subnetzeigentümerrolle abrufen

Um die Subnetzeigentümerrolle abzurufen, ändern Sie folgenden Programmcode:

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

Programmcode: Attribute eines Subnetzeigentümers

Um auf die Subnetze eines Subnetzeigentümers zuzugreifen, verwenden Sie folgenden Programmcode:

```
foreach(Subnet subnet in subnetOwner.Subnets)
{
    Subnet interalSubnet = subnet;
}
```

7.14.4 Typkennung von Subnetzen abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Das Attribut `TypeIdentifier` wird zur Identifizierung eines Subnetzes verwendet. `TypeIdentifier` ist eine Zeichenkette, die aus mehreren Teilen besteht: `<TypeIdentifierType>:<SystemIdentifier>`

Mögliche Werte für `TypeIdentifierType` sind:

- `System`

SystemIdentifier

Subnetztyp	Systemkennung
PROFIBUS	Subnet.Profibus
MPI	Subnet.Mpi
Industrial Ethernet	Subnet.Ethernet

Subnetztyp	Systemkennung
ASI	Subnet.Asi
PtP	Subnet.Ptp
PROFIBUS - integriert	Subnet.ProfibusIntegrated
PC - intern	*null*

Programmcode

Um die Typkennung für vom Benutzer verwaltbare und getrennt erstellbare Objekte für GSD abzurufen, ändern Sie den folgenden Programmcode:

```
Subnet subnet = ...;
string typeIdentifier = subnet.TypeIdentifier;
```

7.14.5 Attribute eines Subnetzes aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein Subnetz bietet bestimmte obligatorische Attribute, die gelesen und/oder geschrieben werden können. Die Attribute sind nur verfügbar, wenn sie in der UI verfügbar sind. Der Schreibvorgang ist im Allgemeinen nur dann zulässig, wenn ein Attribut auch vom Benutzer in der UI geändert werden kann. Dies kann abhängig von der Art des Subnetzes variieren. Der Benutzer kann DpCycleTime nur dann festlegen, wenn IsochronousMode wahr und DpCycleMinTimeAutoCalculation falsch ist.

Attribute von Subnetzen vom Typ ASI

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w		Name des Subnetzes.
NetType	NetType	r		Typ des Subnetzes
SubnetId	String	r	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.

Attribute von Subnetzen vom Typ Ethernet

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w		Name des Subnetzes.
NetType	NetType	r		Typ des Subnetzes
SubnetId	String	r/w	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.
DefaultSubnet	Bool	r/w	dynamisch	Wahr, wenn das Subnetz ein Standardsubnetz ist. In einem Projekt ist höchstens ein Standardsubnetz vorhanden.

Attribute von Subnetzen vom Typ MPI

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w		Name des Subnetzes.
NetType	NetType	r		Typ des Subnetzes
SubnetId	String	r/w	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.
HighestAddress	Int	r/w	dynamisch	Höchste MPI-Adresse am Subnetz.
TransmissionSpeed	BaudRate	r/w	dynamisch	Wahr, wenn das Subnetz ein Standardsubnetz ist. In einem Projekt ist höchstens ein Standardsubnetz vorhanden.

Attribute von Subnetzen vom Typ PC internal

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r		Name des Subnetzes.
NetType	NetType	r		Typ des Subnetzes
SubnetId	String	r	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.

Attribute von Subnetzen vom Typ PROFIBUS

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w		Name des Subnetzes.
NetType	NetType	r		Typ des Subnetzes
SubnetId	String	r/w	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.
HighestAddress	Int	r/w	dynamisch	Höchste PROFIBUS-Adresse am Subnetz.
TransmissionSpeed	BaudRate	r/w	dynamisch	Wahr, wenn das Subnetz ein Standardsubnetz ist. In einem Projekt ist höchstens ein Standardsubnetz vorhanden.
BusProfile	Busprofil	r/w	dynamisch	Das PROFIBUS-Profil.
PbCableConfiguration	Bool	r/w	dynamisch	Wahr, um zusätzliche PROFIBUS-Netzwerkeinstellungen zu aktivieren
PbRepeaterCount	Int	r/w	dynamisch	Anzahl der Repeater für Kupferleitung
PbCopperCableLength	double	r/w	dynamisch	Die Länge der Kupferleitung
PbOpticalComponentCount	Int	r/w	dynamisch	Anzahl der OLMs und OBTs von LWL-Leitung.
PbOpticalCableLength	double	r/w	dynamisch	Die Länge der LWL-Leitung für das PROFIBUS-Netzwerk in km.
PbOpticalRing	Bool	r/w	dynamisch	Wahr, wenn Busparameter für einen optischen Ring übernommen werden
PbOlmP12	Bool	r/w	dynamisch	Wahr, wenn OLM/P12 für die Berechnung des Busparameters aktiviert ist.
PbOlmG12	Bool	r/w	dynamisch	Wahr, wenn OLM/G12 für die Berechnung des Busparameters aktiviert ist.
PbOlmG12Eec	Bool	r/w	dynamisch	Wahr, wenn OLM/G12-EEC für die Berechnung des Busparameters aktiviert ist.
PbOlmG121300	Bool	r/w	dynamisch	Wahr, wenn OLM/G12-1300 für die Berechnung des Busparameters aktiviert ist.
PbAdditionalNetworkDevices	Bool	r/w	dynamisch	Wahr, wenn zusätzliche Busgeräte, die im Projekt nicht vorhanden sind, bei der Berechnung der Buszeiten berücksichtigt werden.
PbAdditionalDpMaster	Int	r/w	dynamisch	Anzahl der nicht konfigurierten DP-Master.
PbTotalDpMaster	Int	r	dynamisch	Gesamtanzahl der DP-Master
PbAdditionalPassiveDevice	Int	r/w	dynamisch	Anzahl der nicht konfigurierten DP-Slaves oder passiven Geräte.
PbTotalPassiveDevice	Int	r	dynamisch	Gesamtanzahl der DP-Slaves oder passiven Geräte.

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
PbAdditionalActiveDevice	Int	r/w	dynamisch	Anzahl der nicht konfigurierten aktiven Geräte mit FDL/FMS/S/-Kommunikationslast.
PbTotalActiveDevice	Int	r	dynamisch	Gesamtanzahl der aktiven Geräte mit FDL/FMS/S/-Kommunikationslast.
PbAdditionalCommunicationLoad	CommunicationLoad	r/w	dynamisch	Grobe Quantifizierung der Kommunikationslast
PbDirectDateExchange	Bool	r/w	dynamisch	Optimierung für direkten Datenaustausch.
PbMinimizeTslotForSlaveFailure	Bool	r/w	dynamisch	Minimierung für Zeitzuweisung für Slave-Fehler.
PbOptimizeCableConfiguration	Bool	r/w	dynamisch	Optimierung der Leitungskonfiguration.
PbCyclicDistribution	Bool	r/w	dynamisch	Wahr, wenn zyklische Verteilung der Busparameter aktiviert ist.
PbTslotInit	Int	r/w	dynamisch	Standardwert von Tslot.
PbTslot	Int	r	dynamisch	Warte-auf-Empfangszeit (slot time)
PbMinTsdr	Int	r/w	dynamisch	Minimale Dauer für die Protokollverarbeitung
PbMaxTsdr	Int	r/w	dynamisch	Maximale Dauer für die Protokollverarbeitung
PbTid1	Int	r	dynamisch	Leerlaufzeit 1
PbTid2	Int	r	dynamisch	Leerlaufzeit 2
PbTrdy	Int	r	dynamisch	Bereitschaftszeit
PbTset	Int	r/w	dynamisch	Einrichtungszeit
PbTqui	Int	r/w	dynamisch	Modulator-Ausklingzeit
PbTtr	int64	r/w	dynamisch	Der Ttr-Wert in t_Bit.
PbTtrTypical	int64	r	dynamisch	Mittlere Antwortzeit am Bus
PbWatchdog	int64	r/w	dynamisch	Ansprechüberwachung
PbGapFactor	Int	r/w	dynamisch	GAP-Aktualisierungsfaktor
PbRetryLimit	Int	r/w	dynamisch	Maximale Anzahl an Wiederholungen
IsochronousMode	Bool	r/w	dynamisch	Wahr, wenn konstante Buszykluszeit aktiviert ist.
PbAdditionalPassivDeviceForIsochronousMode	Int	r/w	dynamisch	Anzahl der zusätzlichen OPs/PGs/TDs usw., die in dieser Netzwerksicht nicht konfiguriert sind.

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
PbTotalPassivDeviceForIsochronousMode	Int	r	dynamisch	Summe der konfigurierten und nicht konfigurierten Geräte, z. B. OPs/PGs/TDs usw.
DpCycleMinTimeAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der kürzesten DP-Zykluszeit aktiviert sind.
DpCycleTime	double	r/w	dynamisch	Die DP-Zykluszeit.
IsochronousTiToAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der Werte IsochronousTi und IsochronousTo aktiviert sind.
IsochronousTi	double	r/w	dynamisch	Time Ti (Zeit zum Lesen von Prozesswerten)
IsochronousTo	double	r/w	dynamisch	Time To (Zeit zum Ausgeben von Prozesswerten)

Attribute von Subnetzen vom Typ PROFIBUS Integrated

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w		Name des Subnetzes.
NetType	NetType	r		Typ des Subnetzes
SubnetId	String	r/w	dynamisch	Eindeutige Kennung des Subnetzes. Die S7-Subnetz-ID setzt sich aus zwei Zahlen zusammen, die durch einen Bindestrich voneinander getrennt werden. Eine Zahl für das Projekt und eine Zahl für das Subnetz, z. B. 4493-1.
IsochronousMode	Bool	r	dynamisch	Konstante Buszykluszeit aktiviert.
DpCycleMinTimeAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der kürzesten DP-Zykluszeit aktiviert sind.
DpCycleTime	double	r/w	dynamisch	Die DP-Zykluszeit.
IsochronousTiToAutoCalculation	Bool	r/w	dynamisch	Wahr, wenn automatische Berechnung und Einstellung der Werte IsochronousTi und IsochronousTo aktiviert sind.
IsochronousTi	double	r/w	dynamisch	Time Ti (Zeit zum Lesen von Prozesswerten)
IsochronousTo	double	r/w	dynamisch	Time To (Zeit zum Ausgeben von Prozesswerten)

Programmcode

Um die Attribute eines Subnetzes abzurufen oder festzulegen, ändern Sie folgenden Programmcode:

```
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");
```

Baud-Raten

Wert	Beschreibung
BaudRate.None	Die Baud-Rate ist unbekannt.
BaudRate.Baud9600	9,6 kBaud
BaudRate.Baud19200	19,2 kBaud
BaudRate.Baud45450	45,45 kBaud
BaudRate.Baud93700	93,75 kBaud
BaudRate.Baud187500	187,5 kBaud
BaudRate.Baud500000	500 kBaud
BaudRate.Baud1500000	1,5 MBaud
BaudRate.Baud3000000	3 MBaud
BaudRate.Baud6000000	6 MBaud
BaudRate.Baud12000000	12 MBaud

Busprofile

Wert	Beschreibung
BusProfile.None	Das Busprofil ist unbekannt.
BusProfile.DP	Der Typ des Netzwerks ist DP.
BusProfile.Standard	Der Typ des Netzwerks ist Standard.
BusProfile.Universal	Der Typ des Netzwerks ist Universal.
BusProfile.UserDefined	Der Typ des Netzwerks ist benutzerdefiniert.

Kommunikationslast

Wert	Beschreibung
CommunicationLoad.None	Keine gültige Kommunikationslast.
CommunicationLoad.Low	Typisch für DP, keine größere Datenkommunikation außer DP.

Wert	Beschreibung
CommunicationLoad.Medium	Typisch für Mischbetrieb von DP und anderen Kommunikationsdiensten (z. B. S7-Kommunikation), wenn DP hohe Zeitanforderungen hat und bei mittlerem azyklischen Kommunikationsaufkommen.
CommunicationLoad.High	Für Mischbetrieb von DP und anderen Kommunikationsdiensten (z. B. S7-Kommunikation), wenn DP geringe Zeitanforderungen hat und bei hohem azyklischen Kommunikationsaufkommen.

7.14.6 Globales Subnetz löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um ein globales Subnetz innerhalb eines Projekts zu löschen, ändern Sie folgenden Programmcode:

```
Project project = ...;
SubnetComposition subnets = projects.Subnets;

// delete subnet
Subnet subnetToDelete = ...;
subnetToDelete.Delete();
```

7.14.7 Alle Beteiligten eines Subnetzes enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Enumeration aller Beteiligten eines Subnetzes.

Programmcode

Um DP-Mastersysteme aus dem Subnetz zu enumerieren, ändern Sie folgenden Programmcode:

```
Subnet subnet = ...;
foreach (Node node in subnet.Nodes)
    {
        // work with the node
    }
```

7.14.8 IO-Systeme eines Subnetzes enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Die Enumeration von IoSystem stellt alle IO-Systeme, die sich auf dem Subnetz befinden, bereit. Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt.

Programmcode

Um DP-Mastersysteme aus dem Subnetz zu enumerieren, ändern Sie folgenden Programmcode:

```
Subnet subnet = ...;
foreach (IoSystem ioSystem in subnet.IoSystems)
    {
        // work with the io system
    }
```

7.14.9 Auf Teilnehmer zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die Rollenschnittstelle aggregiert Teilnehmer: Um auf Attribute zuzugreifen, die mit der Adress- und Subnetzzuordnung einer Schnittstelle zusammenhängen.

Der Name eines Teilnehmers wird in den Attributen einer Schnittstelle im TIA Portal angezeigt. Die Teilnehmerkennung ist eine einzigartige Kennung für jeden auf einer Schnittstelle erfassten Teilnehmer. Der Wert kann nur über TIA Portal Openness angezeigt werden.

Programmcode

Um auf alle Teilnehmer einer Schnittstelle zuzugreifen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (Node node in itf.Nodes)
{
    ... // Work with the node
}
```

Die meisten Schnittstellen haben nur einen einzigen Teilnehmer, daher wird normalerweise der erste Teilnehmer verwendet.

```
NetworkInterface itf = ...
Node node = itf.Nodes.First();
{
    ... // Work with the node
}
```

Teilnehmer stellen ihre Namen und Typen und Teilnehmer-IDs als Attribute zur Verfügung:

```
Node node = ...
string name = node.Name;
NetType type = node.NodeType;
string id = node.NodeId;
```

Netzwerktypen

Wert	Beschreibung
NetType.Unknown	Der Typ des Netzwerks ist unbekannt.
NetType.Ethernet	Der Typ des Netzwerks ist Ethernet.
NetType.Profibus	Der Typ des Netzwerks ist Profibus.
NetType.Mpi	Der Typ des Netzwerks ist MPI.
NetType.ProfibusIntegrated	Der Typ des Netzwerks ist integrierter Profibus.
NetType.Asi	Der Typ des Netzwerks ist ASi.
NetType.PcInternal	Der Typ des Netzwerks ist PC intern.
NetType.Ptp	Der Typ des Netzwerks ist PtP.

7.14.10 Attribute eines Teilnehmers aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein Geräteelement bietet bestimmte obligatorische Attribute, die gelesen und/oder geschrieben werden können. Die Attribute sind nur verfügbar, wenn sie in der UI verfügbar sind. Der Schreibvorgang ist im Allgemeinen nur dann zulässig, wenn ein Attribut auch vom Benutzer in der UI geändert werden kann. Dies kann abhängig von der Art des Geräteelements variieren. Der Benutzer kann RouterAddress nur dann festlegen, wenn RouterUsed wahr ist. Wenn der Benutzer die SubnetMask am IO-Controller ändert, dann wird auch die Subnetzmaske auf allen IO-Devices auf denselben Wert geändert.

Attribute eines Teilnehmers vom Typ ASI

Attribute	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r		Name des Teilnehmers.
NodeId	String	r		ID des Teilnehmers.
NodeType	NetType	r		Ein Teilnehmer ruft seinen Typen vom Subnetz ab.
Address	String	r/w	dynamisch	Weitere Attribute für AS-i-Slaves.

Attribute eines Teilnehmers vom Typ Ethernet

Attribute	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r		Name des Teilnehmers.
NodeId	String	r		ID des Teilnehmers.
NodeType	NetType	r/w oder r		Ein Teilnehmer ruft seinen Typen vom Subnetz ab.
UseIsoProtocol	Bool	r/w	dynamisch	
MacAddress	String	r/w	dynamisch	z. B. 01-80-C2-00-00-00
UseIpProtocol	Bool	r/w	dynamisch	Dieser Wert kann auch dann gelesen werden, wenn er an der entsprechenden TIA UI-Steuerung nicht sichtbar ist.
IpProtocolSelection	Enum	r/w	dynamisch	
Address	String	r/w	dynamisch	nur IPv4 und nicht IPv6 wird unterstützt
SubnetMask	String	r/w	dynamisch	
UseRouter	Bool	r/w	dynamisch	
RouterAddress	String	r/w	dynamisch	
DhcpClientId	String	r/w	dynamisch	
PnDeviceNameSetDirectly	Bool	r/w	dynamisch	PROFINET-Gerätename wird direkt am Gerät festgelegt. Nicht für jedes Gerät verfügbar.
PnDeviceNameAutoGeneration	Bool	r/w	dynamisch	PROFINET-Gerätename wird automatisch erstellt.
PnDeviceName	String	r/w	dynamisch	Eindeutiger Name im Subnetz.
PnDeviceNameConverted	String	r	dynamisch	Gerätename für systeminterne Nutzung umgewandelt.

Attribute eines Teilnehmers vom Typ MPI

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r		Name des Teilnehmers.
NodeId	String	r		ID des Teilnehmers.
NodeType	NetType	r		Ein Teilnehmer ruft seinen Typen vom Subnetz ab.
Address	String	r/w	dynamisch	

Attribute eines Teilnehmers vom Typ PC internal

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r		Name des Teilnehmers.
NodeId	String	r		ID des Teilnehmers.
NodeType	NetType	r		Ein Teilnehmer ruft seinen Typen vom Subnetz ab.

Attribute eines Teilnehmers vom Typ PROFIBUS

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r		Name des Teilnehmers.
NodeId	String	r		ID des Teilnehmers.
NodeType	NetType	r		Ein Teilnehmer ruft seinen Typen vom Subnetz ab.
Address	String	r/w	dynamisch	

Attribute eines Teilnehmers vom Typ PROFIBUS integrated

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r		Name des Teilnehmers.
NodeId	String	r		ID des Teilnehmers.
NodeType	NetType	r		Ein Teilnehmer ruft seinen Typen vom Subnetz ab.
Address	String	r	dynamisch	

Programmcode: Attribute eines Teilnehmers

Um die Attribute eines Teilnehmers abzurufen oder festzulegen, ändern Sie folgenden Programmcode:

```
Node node = ...;
string nameValue = node.Name;
NetType nodeType = node.NodeType;
node.NodeType = NetType.Mpi;
```

Programmcode: Dynamische Attribute

Um dynamische Attribute eines Teilnehmers abzurufen oder festzulegen, ändern Sie folgenden Programmcode:

```
Node node = ...;
var attributeNames = new[]
{
    "Address", "SubnetMask", "RouterAddress", "UseRouter", "DhcpClientId",
"IpProtocolSelection"
};
foreach (var attributeName in attributeNames)
{
    object attributeValue = ((IEngineeringObject)node).GetAttribute(attributeName);
}
```

Protokollauswahl

Wert	Beschreibung
IpProtocolSelection.None	Fehlerwert
IpProtocolSelection.Project	IP Suite innerhalb des Projekts konfiguriert.
IpProtocolSelection.Dhcp	IP Suite mit Hilfe von DHCP-Protokoll verwaltet. ID des DHCP-Client notwendig.
IpProtocolSelection.UserProgram	IP Suite mit Hilfe von FB (Funktionsbaustein) festgelegt.
IpProtocolSelection.OtherPath	IP Suite mit Hilfe von anderen Methoden festgelegt, z. B. PST Tool.
IpProtocolSelection.ViaIoController	IP Suite mit Hilfe von IO-Controller in Runtime festgelegt.

Netzwerktyp

Wert	Beschreibung
NetType.Asi	Netzwerktyp ist ASI.
NetType.Ethernet	Netzwerktyp ist Ethernet.
NetType.Link	Netzwerktyp ist Link.
NetType.Mpi	Netzwerktyp ist MPI.
NetType.PcInternal	Netzwerktyp ist PC internal.
NetType.Profibus	Netzwerktyp ist PROFIBUS.
NetType.ProfibusIntegrated	Netzwerktyp ist PROFIBUS integrated.
NetType.Ptp	Netzwerktyp ist PTP.
NetType.Wan	Netzwerktyp ist Wide Area Network (WAN).
NetType.Unknown	Netzwerktyp ist unbekannt.

7.14.11 Teilnehmer mit einem Subnetz verbinden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um einen Teilnehmer (Gerät, Schnittstelle) einem Netzwerk zuzuweisen, ändern Sie folgenden Programmcode:

```
Node node = ...;
Subnet subnet = ...;
node.ConnectToSubnet(subnet);
```

7.14.12 Teilnehmer von einem Subnetz trennen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um einen Teilnehmer (Gerät, Schnittstelle) von einem Netzwerk zu trennen, ändern Sie folgenden Programmcode:

```
Node node = ...;
node.DisconnectFromSubnet();
```

7.14.13 IO-System erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein IO-System wird durch Aufrufen der Aktion IoController.CreateIoSystem("name") auf einem Objekt vom Typ IoController erstellt. Wenn der name Null oder String.Empty ist, dann wird der Standardname verwendet. Der IO-Controller wird erfasst durch Zugreifen auf das Objekt des Attributs IoControllers an der NetworkInterface. Der Navigator des IoControllers gibt ein IoController-Objekt zurück.

Voraussetzungen für das Erstellen eines IO-Systems:

- Die Schnittstelle des IO-Controllers ist mit einem Subnetz verbunden.
- Der IO-Controller besitzt kein IO-System.

Programmcode

Um ein IO-System zu erstellen, ändern Sie folgenden Programmcode:

```
using System.Linq;
...

NetworkInterface 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 Attribute eines IO-Systems aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt.

Programmcode: Attribute eines IO-Systems

Um die Attribute des IoSystem abzurufen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (IIoController ioController in itf.IoControllers)
{
    IoSystem ioSystem = ioController.IoSystem;
    int ioSystemNumber = ioSystem.Number;
    string ioSystemName = ioSystem.Name;
}
```

Programmcode: Subnetz eines IO-Systems

Um zu dem dem IO-System zugeordneten Subnetz zu navigieren, ändern Sie folgenden Programmcode:

```
Subnet subnet = ioSystem.Subnet;
```

7.14.15 IO-Connector mit einem IO-System verbinden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Verwenden Sie die Aktion ConnectToIoSystem(IoSystem ioSystem) des IoConnector, um einen Profinet- oder DP-IoConnector mit einem vorhandenen IO-System zu verbinden.

Verwenden Sie die Aktion GetIoController, um zum dezentralen IoController zu navigieren. Weitere Informationen dazu, wie Sie zum lokalen IoConnector und dem IO-System navigieren, finden Sie unter Mastersystem oder IO-System einer Schnittstelle abrufen (Seite 200).

Voraussetzungen:

- Der IoConnector ist noch nicht mit einem IO-System verbunden.
- Die IoConnector-Schnittstelle ist mit dem gleichen Subnetz verbunden wie die Schnittstelle des gewünschten IoController.

Programmcode

Ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
IoConnector ioConnector = ...;
ioConnector.ConnectToIoSystem();
IoController ioController = ioConnector.GetIoController();
```

7.14.16 Mastersystem oder IO-System einer Schnittstelle abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Der Dienst NetworkInterface liefert dem Navigator IoControllers, wovon jeder IoController wiederum dem Navigator IoSystem liefert. Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt. Das IO-Device und der Slave werden beide IO-Device genannt.

- Der Navigator des IoControllers gibt IoController-Objekte zurück, wenn die Netzwerkschnittstelle ein IO-System haben kann. Im Moment wird nur ein IO-Controller zurückgegeben.
- Der Navigator des IoConnectors gibt IoConnector-Objekte zurück, wenn die Netzwerkschnittstelle als IO-Gerät mit einem IO-System verbunden werden kann. Im Moment wird nur ein IO-Connector zurückgegeben.

Programmcode: IO-System des IoController abrufen

Um das IO-System des IoController abzurufen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (IoController ioController in itf.IoControllers)
{
    IoSystem ioSystem = ioController.IoSystem;
    // work with the io system
}
```

Programmcode: IO-System des IoConnector abrufen

Um das IO-System des IoConnector abzurufen, ändern Sie folgenden Programmcode:

```
NetInterface itf = ...
foreach (IoConnector ioConnector in itf.IoConnectors)
{
    IoSystem ioSystem = ioConnector.ConnectedIoSystem;
    // work with the io system
}
```

7.14.17 IO-Controller abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Derzeit sind nur Konfigurationen mit einem IoController möglich. Ein IoController verfügt nicht über modellierte Attribute oder Aktionen.

Programmcode

Um den IO Controller abzurufen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (IoController ioController in itf.IoControllers)
{
    // work with the io controller
}
```

7.14.18 IO-Connector abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein IoConnector verfügt nicht über modellierte Attribute oder Aktionen.

Programmcode

Um den IO Connector abzurufen, ändern Sie folgenden Programmcode:

```
NetworkInterface itf = ...
foreach (IoConnector ioConnector in itf.IoConnectors)
{
    // work with the IoConnector
}
```

7.14.19 IO-Connector von einem IO-System oder einem DP-Mastersystem trennen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Verwenden Sie die Aktion DisconnectFromIoSystem() des IoConnector, um einen IoConnector von einem vorhandenen IO-System oder einem vorhandenen DP-Mastersystem zu trennen.

Weitere Informationen dazu, wie Sie zum lokalen IoConnector und dem IO-System navigieren, finden Sie unter Mastersystem oder IO-System einer Schnittstelle abrufen (Seite 200).

Programmcode

Ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
IoConnector ioConnector = ...;

ioConnector.DisconnectFromIoSystem();
```

7.14.20 Attribute eines DP-Mastersystems aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein DP-Mastersystem bietet bestimmte Attribute, die gelesen und/oder geschrieben werden können. Die Attribute sind nur verfügbar, wenn sie in der UI verfügbar sind. Der Schreibvorgang ist im Allgemeinen nur dann zulässig, wenn ein Attribut auch vom Benutzer in der UI geändert werden kann. Dies kann abhängig vom DP-Master und den DP-Slaves variieren, die diesem DP-Mastersystem zugeordnet werden.

Attribute eines DP-Mastersystems

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
Name	String	r/w		
Number	Int	r/w		Das Attribut Number übernimmt Werte, die nicht über die UI festgelegt werden können. In einem solchen Fall schlägt die Übersetzung fehl.

Programmcode: Attribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
IoSystem dpMastersystem = ...;

string name = dpMastersystem.Name;
int number = dpMastersystem.Number;
```

Programmcode: Attribute festlegen

Um die Attribute einzustellen, ändern Sie folgenden Programmcode:

```
IoSystem dpMastersystem = ...;

dpMastersystem.Name ="myDpMastersystem"
dpMastersystem.Number=42;
```

7.14.21 Attribute eines PROFINET IO-Systems aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein IO-System bietet bestimmte Attribute, die gelesen und/oder geschrieben werden können. Die Attribute sind nur verfügbar, wenn sie in der UI verfügbar sind. Der Schreibvorgang ist im Allgemeinen nur dann zulässig, wenn ein Attribut auch vom Benutzer in der UI geändert werden kann. Dies kann abhängig vom IO-Controller und den IO-Devices variieren, die diesem IO-System zugeordnet werden.

Attribute eines PROFINET IO-Systems

Attribut	Datentyp	Schreibbar	Zugriff	Beschreibung
MultipleUseIoSystem	Bool	r/w	dynamisch	
Name	String	r/w		
Number	Int	r/w		Das Attribut Number übernimmt Werte, die nicht über die UI festgelegt werden können. In einem solchen Fall schlägt die Übersetzung fehl.
UseIoSystemNameAsDeviceNameExtension	Bool	r/w	dynamisch	Wenn MultipleUseIoSystem auf TRUE gesetzt ist, dann wird UseIoSystemNameAsDeviceNameExtension auf FALSE gesetzt und der Schreibzugriff ist nicht möglich.
MaxNumberIWlanLinksPerSegment	Int	r/w	dynamisch	

Programmcode: Attribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
string name = ioSystem.Name;
```

Programmcode: Attribute festlegen

Um die Attribute einzustellen, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
ioSystem.Name = "IOSystem_1";
```

Programmcode: Attribute bei dynamischem Zugriff abrufen

Um die Werte von dynamischen Attributen abzurufen, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
var attributeNames = new[]
{
    "MultipleUseIoSystem", "UseIoSystemNameAsDeviceNameExtension",
"MaxNumberIWlanLinksPerSegment"
};
foreach (var attributeName in attributeNames)
{
    object attributeValue = ((IEngineeringObject)ioSystem).GetAttribute(attributeName);
}
```

Programmcode: Attribute bei dynamischem Zugriff festlegen

Um die Werte der dynamischen Attribute festzulegen, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
((IEngineeringObject)ioSystem).SetAttribute("MultipleUseIoSystem", true);
```

7.14.22 DP-Mastersystem löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode: PROFINET IO-System löschen

Um ein PROFINET IO-System zu löschen, ändern Sie den folgenden Programmcode:

```
IoController ioController = ...;
IoSystem ioSystem = ioController.IoSystem;

ioSystem.Delete();
```

7.14.23 Profinet IO-System löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um ein Profinet IO-System zu löschen, ändern Sie den folgenden Programmcode:

```
IoController ioController = ...;
IoSystem ioSystem = ioController.IoSystem;
ioSystem.Delete();
```

7.14.24 DP-Mastersystem erstellen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein DP-Mastersystem wird durch Aufrufen der Aktion CreateIoSystem(string nameOfIoSystem) auf einem Objekt vom Typ IoController erstellt. Der IO-Controller wird erfasst durch Zugreifen auf das Objekt des Attributs IoControllers an der NetworkInterface.

Voraussetzungen für das Erstellen eines DP-Mastersystems:

- Die Schnittstelle des IO-Controllers ist mit einem Subnetz verbunden.
- Der IO-Controller besitzt kein IO-System.

Programmcode

Um ein DP-Mastersystem zu erstellen, ändern Sie folgenden Programmcode:

```
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 Port-Verschaltungsinformationen des Port-Geräteelements aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

NetworkPort bietet die Verknüpfung ConnectedPorts, die eine Enumeration von Ports ist, um auf alle verschalteten Partnerports eines Ports zuzugreifen.

Es können nur Ports miteinander verschaltet werden, die auch in der TIA UI verschaltet werden können. Beispielsweise können zwei Ports derselben Ethernet-Schnittstelle nicht miteinander verschaltet werden. Eine wiederherstellbare Ausnahme wird ausgelöst,

- wenn bereits eine Verschaltung mit demselben Partnerport besteht
- wenn Sie versuchen, zwei Ports zu verschalten, die nicht miteinander verschaltet werden können
- wenn Sie versuchen, eine zweite Verschaltung zu einem Port anzulegen, der keine alternativen Partner unterstützt

Programmcode: Portverschaltung abrufen

Um die Port-Verschaltungsinformationen eines Port-Geräteelements abzurufen, ändern Sie den folgenden Programmcode:

```
NetworkPort port = ...;
foreach (NetworkPort partnerPort in port.ConnectedPorts)
{
    // work with the partner port
}
```

Programmcode: Portverschaltungen erstellen

Ändern Sie folgenden Programmcode:

```
NetworkPort port1 = ...;
NetworkPort port2 = ...;
port1.ConnectToPort(port2);

// port supports alternative partners
NetworkPort port1 = ...;
NetworkPort port2 = ...;
NetworkPort port3 = ...;
port1.ConnectToPort(port2);
port1.ConnectToPort(port3);
```

Programmcode: Portverschaltung löschen

Ändern Sie folgenden Programmcode:

```
NetworkPort port1 = ...;
NetworkPort port2 = ...;
port1.DisconnectFromPort(port2);
```

7.14.26 Attribute der Port-Verschaltung

Attribute für Port-Verschaltungen

TIA Portal Openness unterstützt die folgenden Attribute für Port-Verschaltungen. Wenn die Attribute in der Benutzeroberfläche verfügbar sind, sind die entsprechenden Attribute auch über TIA Portal Openness zugänglich. Wenn der Anwender Zugriff hat und die Attribute in der Benutzeroberfläche ändern kann, können die Attribute auch über TIA Portal Openness geändert werden.

Attributname	Datentyp	Schreibbar	Zugriff	Beschreibung
MediumAttachment-Type	MediumAttachment-Type	Schreibgeschützt	Dynamisches Attribut	
CableName	CabelName	Lesen-Schreiben	Dynamisches Attribut	
AlternativePartnerP-orts	Bool	Lesen-Schreiben	Dynamisches Attribut	Nur verfügbar, wenn die Funktionalität des Werkzeugwechslers unterstützt wird.
SignalDelaySelection	SignalDelaySelection	Lesen-Schreiben	Dynamisches Attribut	
CableLength	CableLength	Lesen-Schreiben	Dynamisches Attribut	
SignalDelayTime	Double	Lesen-Schreiben	Dynamisches Attribut	

Enum-Werte von Port-Verschaltungsattributen

Die Enum MediumAttachmentType hat folgende Werte.

Wert	Beschreibung
MediumAttachmentType.None	Anbindungstyp kann nicht ermittelt werden.
MediumAttachmentType.Copper	Anbindungstyp ist Kupfer.
MediumAttachmentType.FiberOptic	Anbindungstyp ist Glasfaser.

Die Enum CableName hat folgenden Wert

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

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

Die Enum SignalDelaySelection hat folgende Werte.

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

Die Enum CableLength hat folgende Werte

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

Attribute von Port-Optionen

Die Attribute von Port-Optionen werden nachfolgend dargestellt.

Attributname	Datentyp	Schreibbar	Zugriff
PortActivation	Bool	Lesen-Schreiben	Dynamisches Attribut
TransmissionRateAndDuplex	TransmissionRateAndDuplex	Lesen-Schreiben	Dynamisches Attribut
PortMonitoring	Bool	Lesen-Schreiben	Dynamisches Attribut
TransmissionRateAutoNegotiation	Bool	Lesen-Schreiben	Dynamisches Attribut
EndOfDetectionOfAccessibleDevices	Bool	Lesen-Schreiben	Dynamisches Attribut
EndOfTopologyDiscovery	Bool	Lesen-Schreiben	Dynamisches Attribut
EndOfSyncDomain	Bool	Lesen-Schreiben	Dynamisches Attribut

Die Enum TransmissionRateAndDuplex hat folgende Werte.

Wert	Beschreibung
TransmissionRateAndDuplex.None	
TransmissionRateAndDuplex.Automatic	Automatisch
TransmissionRateAndDuplex.AUI10Mbps	10 Mbps AUI
TransmissionRateAndDuplex.TP10MbpsHalfDuplex	TP 10 Mbps Halbduplex
TransmissionRateAndDuplex.TP10MbpsFullDuplex	TP 10 Mbps Vollduplex
TransmissionRateAndDuplex.AsyncFiber10MbpsHalfDuplex	Asynchrone Glasfaser 10 Mbit/s Halbduplexmodus
TransmissionRateAndDuplex.AsyncFiber10MbpsFullDuplex	Asynchrone Glasfaser 10 Mbit/s Vollduplexmodus
TransmissionRateAndDuplex.TP100MbpsHalfDuplex	TP 100 Mbps Halbduplex
TransmissionRateAndDuplex.TP100MbpsFullDuplex	TP 100 Mbps Vollduplex
TransmissionRateAndDuplex.FO100MbpsFullDuplex	FO 100 Mbps Vollduplex
TransmissionRateAndDuplex.X1000MbpsFullDuplex	X1000 Mbps Vollduplex
TransmissionRateAndDuplex.FO1000MbpsFullDuplexLD	FO 1000 Mbps Vollduplex LD
TransmissionRateAndDuplex.FO1000MbpsFullDuplex	FO 1000 Mbps Vollduplex
TransmissionRateAndDuplex.TP1000MbpsFullDuplex	TP 1000 Mbps Vollduplex
TransmissionRateAndDuplex.FO10000MbpsFullDuplex	FO 10000 Mbps Vollduplex
TransmissionRateAndDuplex.FO100MbpsFullDuplexLD	FO 100 Mbps Vollduplex LD
TransmissionRateAndDuplex.POFPCF100MbpsFullDuplexLD	POF/PCF 100 Mbps Vollduplex

7.14.27 Attribute eines Ports aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein Geräteelement, das zugleich ein Port ist, bietet gegenüber einem einfachen Geräteelement zusätzliche Funktionalität.

- Es ist möglich, auf die verknüpften Partnerports des Ports zuzugreifen.
- Es ist möglich, auf die Schnittstelle des Ports zuzugreifen.

Um auf diese zusätzliche Funktionalität zuzugreifen, muss die NetworkPort-Funktion, ein bestimmter Dienst des Geräteelements, verwendet werden.

Programmcode: Auf einen Port zugreifen

Um auf Attribute eines Kanals zuzugreifen, ändern Sie folgenden Programmcode:

```
NetworkPort port = ((IEngineeringServiceProvider)deviceItem).GetService<NetworkPort>();
if (port != null)
{
    ... // Work with the port
}
```

Attribute eines Ports

Ein Port hat folgende Attribute:

```
NetworkPort port = ...;
var connectedPorts = port.ConnectedPorts;
var myInterface = port.Interface;
```

7.14.28 DP-Mastersysteme eines Subnetzes enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Die Enumeration von IoSystem stellt alle DP-Mastersysteme, die sich auf dem Subnetz befinden, bereit. Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt.

Programmcode

Um DP-Mastersysteme aus dem Subnetz zu enumerieren, ändern Sie folgenden Programmcode:

```
Subnet subnet = ...;
foreach (IoSystem ioSystem in subnet.IoSystems)
    {
        // work with the io system
    }
```

7.14.29 Zugeordnete IO-Connectors enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Das Mastersystem und das IO-System werden beide von der Klasse IoSystem dargestellt.

Wird verwendet für:

- Enumeration zugeordneter IO-Connectors des DP-Mastersystems
- Enumeration zugeordneter IO-Connectors des Profinet-IO-Systems

Programmcode

Um zugeordnete IO Connectors des DP-Mastersystems zu enumerieren, ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
foreach (IoConnector ioConnector in ioSystem.ConnectedIoDevices)
    {
        // work with the io connector
    }
```

7.14.30 DP-IO-Connector mit einem DP-Mastersystem verbinden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Verwenden Sie die Aktion ConnectToIoSystem(IoSystem ioSystem) des IoConnector, um einen IoConnector mit einem vorhandenen DP-Mastersystem zu verbinden.

Verwenden Sie die Aktion GetIoController, um zum dezentralen IoController zu navigieren. Weitere Informationen dazu, wie Sie zum lokalen IoConnector und dem IO-System navigieren, finden Sie unter Mastersystem oder IO-System einer Schnittstelle abrufen (Seite 200).

Voraussetzungen:

- Der IoConnector ist noch nicht mit einem IO-System verbunden.
- Die IoConnector-Schnittstelle ist mit dem gleichen Subnetz verbunden wie die Schnittstelle des gewünschten IoController.

Programmcode

Ändern Sie folgenden Programmcode:

```
IoSystem ioSystem = ...;
IoConnector ioConnector = ...;
ioConnector.ConnectToIoSystem(ioSystem);
IoController ioController = ioConnector.GetIoController();
```

7.15 Funktionen auf Geräten

7.15.1 Obligatorische Attribute von Geräten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Jedes Gerät oder Geräteelement besitzt bestimmte obligatorische Attribute, die gelesen und/oder geschrieben werden können. Diese Attribute sind immer die gleichen wie in der Benutzeroberfläche des TIA Portals.

In Openness werden die folgenden Attribute unterstützt:

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
Author	String	read/write	dynamisch	
Comment	String	read/write	dynamisch	manchmal schreibgeschützter Zugriff
CommentML	MultilingualTextItem	read/write	dynamisch	manchmal schreibgeschützter Zugriff
IsGsd	Bool	read		WAHR, wenn die Gerätebeschreibung über GSD/GSDML installiert wird
Name	String	read/write		manchmal schreibgeschützter Zugriff
TypeIdentifier	String	read		
TypeName	String	read	dynamisch	

Programmcode: Obligatorische Attribute eines Geräts

Ändern Sie den folgenden Programmcode, um die obligatorischen Attribute eines Geräts abzurufen:

```
Device device = ...;
string nameValue = device.Name;
bool isGsdValue = device.IsGsd;
```

Programmcode: Obligatorische Attribute bei dynamischem Zugriff

Ändern Sie den folgenden Programmcode, um die Attribute bei dynamischem Zugriff aufzurufen:

```
Device device = ...;
var attributeNames = new[] {
    "TypeName", "Author", "Comment"
    };
foreach (var attributeName in attributeNames) {
    object attributeValue = ((IEngineeringObject)device).GetAttribute(attributeName);
    }
```

7.15.2 Typkennung von Geräten und Geräteelementen abrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Das Attribut `TypeIdentifier` wird zur Identifizierung eines Hardwareobjekts verwendet, das über TIA Portal Openness API erstellt werden kann. `TypeIdentifier` ist eine Zeichenkette, die aus mehreren Teilen besteht: `<TypeIdentifierType>:<Identifier>`

Mögliche Werte für `TypeIdentifierType` sind:

- `OrderNumber`
- `GSD`
- `System`

OrderNumber

`OrderNumber` ist die allgemeine `TypeIdentifier` für alle im Hardware-Katalog vorhandenen Module.

Format der Typkennung	Beispiel	Besonderheiten
`<OrderNumber>`	OrderNumber:3RK1 200-0CE00-0AA2	
`<OrderNumber>/<FirmwareVersion>`	OrderNumber:6ES7 510-1DJ01-0AB0/V2.0	Die Firmware-Version ist optional, wenn diese gar nicht oder nur eine Version im System vorhanden ist. Bitte beachten
`<OrderNumber>//<AdditionalTypeIdentifier>`	OrderNumber:6AV2 124-2DC01-0AX0//Landscape	Die zusätzliche Typkennung kann notwendig sein, wenn `OrderNumber` und `FirmwareVersion` keinen eindeutigen Treffer im System haben.

Hinweis

Es gibt einige wenige Module im Hardware-Katalog, die Platzhalterzeichen in der Bestellnummer enthalten, welche eine bestimmte Gruppe realer Hardware repräsentiert, z. B. die verschiedenen Längen der S7-300 Baugruppenträger. In diesem Fall können sowohl die `OrderNumber` als auch die Platzhalter-`OrderNumber` verwendet werden, um eine Instanz des Hardwareobjekts zu erstellen. Platzhalter können jedoch nicht beliebig an irgendeiner Stelle verwendet werden.

GSD

Dies ist die Kennung, die für Module verwendet wird, die über GSD oder GSDML dem TIA Portal hinzugefügt werden.

Format der Typkennung	Beispiel	Besonderheiten
`<GsdName>/<GsdType>`	GSD:SIEM8139.GSD/DAP	`GsdName` ist der Name des GSD oder GSDML in Großbuchstaben. `GsdType` ist eines der folgenden: • D: Gerät (device) • R: Baugruppenträger (rack) • DAP: Kopfmodul • M: Modul • SM: Submodul GsdId ist die Typkennung.
`<GsdName>/<GsdType>/<GsdId>`	GSD:SIEM8139.GSD/M/4	

System

Dies ist die Kennung für Objekte, die nicht mittels `OrderNumber` oder `GSD` bestimmt werden können.

Format der Typkennung	Beispiel	Besonderheiten
`<SystemTypeIdentifier>`	System:Device.S7300	`SystemTypeIdentifier` ist die primäre Kennung eines Objekts. `AdditionalTypeIdentifier` kann notwendig sein, wenn die `SystemTypeIdentifier` nicht eindeutig ist. Die Präfixe für bestimmte Objekttypen sind: • `Connection.` • `Subnet.` • `Device.` • `Rack.`
`<SystemTypeIdentifier>/` `<AdditionalTypeIdentifier>`	GSD:SIEM8139.GSD/M/4	

Programmcode

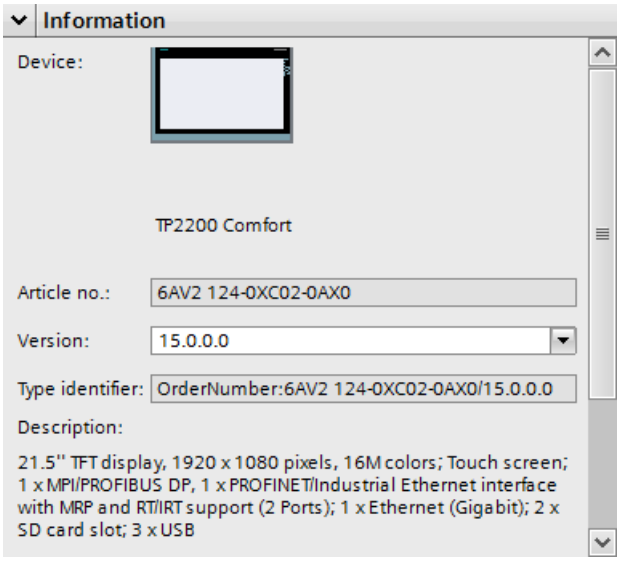
Um die Typkennung für vom Benutzer verwaltbare und getrennt erstellbare Objekte für GSD abzurufen, ändern Sie den folgenden Programmcode:

```
HardwareObject hardwareObject = ...;
string typeIdentifier = hardwareObject.TypeIdentifier;
```

Typkennungen im TIA Portal anzeigen

Wenn Sie eine Typkennung kennen müssen, ermitteln Sie sie im TIA Portal wie folgt:

1. Aktivieren Sie unter "Optionen > Einstellungen > Hardware-Konfiguration > Anzeige der Typkennung" die Einstellung "Anzeige der Typkennung für Geräte und Module".
2. Öffnen Sie den Editor "Geräte & Netze".
3. Wählen Sie ein Gerät im Katalog aus.
   Die Typkennung wird unter "Information" angezeigt.



7.15.3 Erstellen eines Geräts

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein Gerät kann auf zwei Arten erstellt werden, in einem Projekt oder in einer Gerätegruppe:

- Erstellen eines Geräts über eine Geräteelement-Typkennung wie im TIA-Hardwarekatalog
  `Device CreateWithItem(DeviceItemTypeId, DeviceItemName, DeviceName)`
- Nur Gerät erstellen
  `Device Create(DeviceTypeId, DeviceName)`

Name	Typ	Beschreibung
DeviceItemTypeId	String	Typkennung des Geräteelements
DeviceTypeId	String	Typkennung des Geräts
DeviceItemName	String	Name des erstellten Geräteelements
DeviceName	String	Name des erstellten Geräts

Siehe: Typkennung (Seite 216)

Programmcode: Gerät mit Typkennung erstellen

Um ein Geräteobjekt über eine Typkennung zu erstellen, ändern Sie den folgenden Code:

```
DeviceComposition devices = ...;
Device device = devices.CreateWithItem("OrderNumber:6ES7 510-1DJ01-0AB0/V2.0", "PLC_1",
"NewDevice");
Device gsdDevice = devices.CreateWithItem("GSD:SIEM8139.GSD/M/4 ", "GSD Module",
"NewGsdDevice");
```

Programmcode: Nur Gerät erstellen

Um nur das Geräteobjekt zu erstellen, ändern Sie den folgenden Code:

```
DeviceComposition devices = ...;
Device deviceOnly = devices.Create("System:Device.S7300", "S7300Device");
Device gsdDeviceOnly = devices.Create("GSD:SIEM8139.GSD/D", "GSD Device");
```

7.15.4 Geräte enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung: Geräte enumerieren

Die TIA Portal Openness API ordnet Geräte ähnlich der Projektnavigation im TIA Portal an. PNV.

- Geräte, die dem Projekt direkt untergeordnet sind, werden in der Zusammensetzung "Devices" des Projekts gesammelt angeordnet.
- Geräte, die sich in Geräteordnern befinden, werden in der Zusammensetzung "Devices" des Ordners gesammelt angeordnet.

Hinweis

Hierarchie von Hardware-Objekten des Objektmodells (Seite 64) beachten.

Um die Geräte eines Projekts zu enumerieren, verwenden Sie eine der folgenden Möglichkeiten:

- Alle Geräte auf erster Ebene enumerieren
- Alle Geräte in Gruppen oder Untergruppen enumerieren
- Alle Geräte eines keine Gerätegruppen enthaltenden Projekts enumerieren
- Alle Geräte ungruppierter Gerätesystemgruppen enumerieren

Beispiele für enumerierbare Geräte:

- Central station
- PB-Slave / PN-IO device
- HMI Device

Programmcode: Geräte auf erster Ebene enumerieren

Um Geräte auf erster Ebene zu enumerieren, ändern Sie den folgenden Programmcode:

```
private static void EnumerateDevicesInProject(Project project)
{
    DeviceComposition deviceComposition = project.Devices;
    foreach (Device device in deviceComposition)
    {
    // add code here
    }
}
```

Ändern Sie den folgenden Programmcode, um auf ein einzelnes Gerät zuzugreifen:

```
private static void AccessSingleDeviceByName(Project project)
{
    DeviceComposition deviceComposition = project.Devices;
    // The parameter specifies the name of the device
    Device device = deviceComposition.Find("MyDevice");
}
```

Programmcode: Geräte in Gruppen oder Untergruppen enumerieren

Um auf Geräte in einer Gruppe zuzugreifen, müssen Sie zuerst zu der Gruppe und danach zu dem Gerät navigieren.

Ändern Sie folgenden Programmcode:

```
//Enamerate devices in groups or sub-groups
private static void EnumerateDevicesInGroups(Project project)
{
    foreach (DeviceUserGroup deviceUserGroup in project.DeviceGroups)
    {
        EnumerateDeviceUserGroup(deviceUserGroup);
    }
}
private static void EnumerateDeviceUserGroup(DeviceUserGroup deviceUserGroup)
{
    EnumerateDeviceObjects(deviceUserGroup.Devices);
    foreach (deviceUserGroup subDeviceUserGroup in deviceUserGroup.Groups)
    {
        // recursion
        EnumerateDeviceUserGroup(subDeviceUserGroup);
    }
}
private static void EnumerateDeviceObjects(DeviceComposition deviceComposition)
{
    foreach (Device device in deviceComposition)
    {
    // add code here
    }
}
```

Programmcode: Spezifische Geräte finden

Zum Finden eines bestimmten Geräts unter seinem Namen ändern Sie den folgenden Programmcode:

```
//Find a specific device by name
Project project = ...
Device plc1 = project.Devices.First(d => d.Name == "Mydevice");
... // Work with the device
```

Zum Finden eines bestimmten Geräts über die Methode "Find" ändern Sie den folgenden Programmcode:

```
//Find a specific device via "Find" method
Project project = ...
Device plc1 = project.Devices.Find("MyDevice");
... // Work with the device
```

Programmcode: Geräte eines keine Gerätegruppen enthaltenden Projekts enumerieren

Ändern Sie folgenden Programmcode:

```
//Enumerate all devices which are located directly under a project that contains no device
groups
Project project = ...
foreach (Device device in project.Devices)
{
    ... // Work with the devices
}
```

Programmcode: Alle Geräte in einem Ordner enumerieren

Ändern Sie folgenden Programmcode:

```
//Enumerate all devices located in a folder
Project project = ...
DeviceUserGroup sortingGroup = project.DeviceGroups.Find ("Sorting");
Device plc1 = sortingGroup.Devices.First(d => d.Name == "MyStationName");
... // Work with the device
```

Programmcode: Geräte ungruppierter Gerätesystemgruppen enumerieren

Zum Strukturieren der Projekte wurden dezentrale Geräte in die Gruppe UngroupedDevices eingefügt. Um auf diese Gruppe zuzugreifen, navigieren Sie zuerst zur Gruppe und danach zum Gerät.

Ändern Sie folgenden Programmcode:

```
//Enumerate devices of the ungrouped device system group
Project project = ...
DeviceSystemGroup group = project.UngroupedDevicesGroup;
Device plc1 = group.Devices.First(d => d.Name == "MyStationName");
... // Work with the device
```

7.15.5 Auf Geräte zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Jedes GSD- oder GSDML-basierte Gerät verfügt über Attribute. Einige davon dienen zur Ermittlung des genauen Typs des Geräts.

Name	Datentyp	Schreibbar	Zugriff	Beschreibung
Author	String	read/write	dynamisch	
Comment	String	read/write	dynamisch	
GsdName	String	read	dynamisch	Name der GSD- oder GSDML-Datei.
GsdType	String	read	dynamisch	Typ des Hardwareobjekts. Der Wert des Geräts ist stets "D".
GsdId	String	read	dynamisch	Spezifische Kennung des Hardwareobjekts. Für Geräte immer leer.
IsGsd	Bool	read		WAHR für GSD-Gerät oder GSDML-Gerät
Name	String	read/write		
TypeIdentifier	String	read		

Programmcode: Identifikationsattribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId"
    ;
foreach (var attributeName in attributeNames) {
    object attributeValue = device.GetAttribute(attributeName);
    }
```

Programmcode: Attribute

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Device device = ...;
string nameValue = device.Name;
bool isGsdValue = device.IsGsd;
```

Programmcode: Attribute bei dynamischem Zugriff

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId"
    ;
foreach (var attributeName in attributeNames) {
    object attributeValue = device.GetAttribute(attributeName);
    }
```

Besonderheiten von GSD-Geräten

Ein Gerät, das zugleich ein GSD-Gerät ist, bietet zusätzliche Funktionalität. Um die Funktion GsdDevice abzurufen, verwenden Sie die Methode GetService.

```
GsdDevice gsdDevice = ((IEngineeringServiceProvider)deviceItem).GetService<GsdDevice>();
if (gsdDevice != null) {
    ... // work with the GSD device
    };
```

Programmcode: Attribute eines GSD-Geräts

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
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 Löschen eines Geräts

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Ändern Sie den folgenden Programmcode, um ein Gerät zu löschen:

```
Project project = ...;
Device deviceToDelete = project.UngroupedDevices.Devices.Find("......");

// delete device
deviceToDelete.Delete();
```

7.16 Funktionen auf Geräteelementen

7.16.1 Obligatorische Attribute von Geräteelementen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Jedes Gerät oder Geräteelement besitzt bestimmte obligatorische Attribute, die gelesen und/oder geschrieben werden können. Diese Attribute sind immer die gleichen wie in der Benutzeroberfläche des TIA Portals.

In TIA Portal Openness werden die folgenden Attribute unterstützt:

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
Author	String	read/write	dynamisch	
Classification	DeviceItemClassification	read		
Comment	String	read/write	dynamisch	manchmal schreibgeschützter Zugriff
CommentML	MultilingualTextItem	read/write	dynamisch	manchmal schreibgeschützter Zugriff
FirmwareVersion	String	read	dynamisch	
InterfaceOperatingMode	InterfaceOperatingModes	read/write	dynamisch	Für Geräteelemente mit der Funktion `NetworkInterface`
InterfaceType	NetType	readwrite	dynamisch	Für Geräteelemente mit der Funktion `NetworkInterface`
IsBuiltIn	Bool	read		FALSCH für Objekte, die durch den Anwender erstellt werden können
IsGsd	Bool	read		WAHR, wenn die Gerätebeschreibung über GSD/GSDML installiert wird
IsPlugged	Bool	read		WAHR für gesteckte Geräte
Label	String	read	dynamisch	Für Geräteelemente mit der Funktion `NetworkPort` oder `NetworkInterface`. Wenn die Schnittstelle oder der Port kein "Label" hat, dann ist `Label` gleich `String.Empty`.
LocationIdentifier	String	read/write	dynamisch	
Name	String	read/write		manchmal schreibgeschützter Zugriff

Attributname	Datentyp	Schreib-bar	Zugriff	Kommentar
OrderNumber	String	read/write	dyna-misch	manchmal schreibgeschützter Zugriff
PlantDesignation	String	read/write	dyna-misch	
PositionNumber	int	read		
TypeIdentifier	String	read		
TypeName	String	read	dyna-misch	Der sprachenunabhängige Typname. Optional für Geräte-elemente, die vom Anwender als automatisch-erstellte Ele-mente oder feste Submodule nicht verwaltbar sind).

Klassifikation des Geräteelements

Wert	Beschreibung
DeviceItemClassifications.None	Keine Klassifikation.
DeviceItemClassifications.CPU	Das Geräteelement ist eine CPU
DeviceItemClassifications.HM	Das Geräteelement ist ein Kopfmodul.

Programmcode: Obligatorische Attribute eines Gerätelements

Ändern Sie den folgenden Programmcode, um die obligatorischen Attribute eines Geräteelements abzurufen:

```
DeviceItem deviceItem = ...;
string nameValue = deviceItem.Name;
string typeIdentfierValue = deviceItem.TypeIdentifier;
int positionNumberValue = deviceItem.PositionNumber;
bool isBuiltInValue = deviceItem.IsBuiltIn;
bool isPluggedValue = deviceItem.IsPlugged;
```

Programmcode: Obligatorische Attribute bei dynamischem Zugriff

Ändern Sie den folgenden Programmcode, um die Attribute bei dynamischem Zugriff aufzurufen:

```
Device device = ...;
var attributeNames = new[] {
    "TypeName", "Author", "Comment", "OrderNumber", "FirmwareVersion", "PlantDesignation",
"LocationIdentifier"
    };
foreach (var attributeName in attributeNames) {
    object attributeValue = ((IEngineeringObject)deviceItem).GetAttribute(attributeName);
    }

DeviceItem deviceItem = ...;
((IEngineeringObject)deviceItem).SetAttribute("Comment", "This is a comment.");
```

7.16.2 Ein Geräteelement erstellen und stecken

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Die Aktion `PlugNew(string typeIdentifier, string name, int positionNumber)` von `HardwareObject` wird verwendet, um

- ein neues Geräteelement zu erstellen und in ein vorhandenes Hardwareobjekt zu stecken
- ein neues Untergeräteelement zu erstellen, z. B. ein Submodul, und dieses in ein Geräteelement zu stecken

Wenn die Aktion erfolgreich war, gibt sie das erstellte Geräteelementobjekt aus. Andernfalls wird eine wiederherstellbare Ausnahme ausgelöst.

Mit der Aktion `CanPlugNew(string typeIdentifier, string name, int positionNumber)` können Sie ermitteln, ob Erstellen und Stecken möglich sind. Wenn eine Ausführung nicht möglich ist, gibt die Aktion `false` aus.

Wenn die Methode den Wert True zurückgibt, kann die Aktion aus den folgenden unvorhersehbaren Gründen dennoch fehlschlagen.

- eine Positionsnummer wird bereits von einem anderen Geräteelement verwendet
- das aktuelle Geräteelement kann nicht in die Position gesteckt werden, obwohl sie frei ist
- der Behälter stellt die Positionsnummer nicht bereit
- der Name des Geräteelements wird bereits von einem vorhandenen Geräteelement im selben Behälter verwendet
- das Geräteelement kann nicht in den Behälter gesteckt werden
- das Gerät ist online

Die folgende Tabelle zeigt die benötigten Methodenparameter:

Name	Typ	Beschreibung
`typeIdentifier`	String	Typkennung des erstellten Geräteelements
`name`	String	Name des erstellten Geräteelements
`positionNumber`	int	Positionsnummer des erstellten Geräteelements

Programmcode

Um ein Geräteelement in ein vorhandenes Hardwareobjekt zu stecken, ändern Sie den folgenden Programmcode:

```
HardwareObject hwObject = ...;
string typeIdentifier = ...;
string name = ...;
int positionNumber = ...;
if(hwObject.CanPlugNew(typeIdentifier, name, positionNumber))
{
    DeviceItem newPluggedDeviceItem = hwObject.PlugNew(typeIdentifier, name,
positionNumber);
}
```

Auf Modulinformationen zugreifen

Der TIA Portal Openness-Anwender kann über das Objekt ModuleInformationProvider auf Informationen zu den steckbaren Modulen zugreifen. Der Anwender hat Zugriff auf

- die Behältertypen, in die ein bestimmtes Modul zu stecken ist (z. B. Gerät und Baugruppenträger) – über die Methode FindContainerTypes.
- die verfügbaren Versionen für ein bestimmtes teilweise angegebenes Modul – über die Methode FindModuleTypes.

Programmcode: Auf das Objekt ModuleInformationProvider zugreifen

```
Project project = ...;
HardwareUtilityComposition extensions = project.HwUtilities;
var result = extensions.Find("ModuleInformationProvider") as ModuleInformationProvider;
```

Programmcode: Mit der Methode FindContinerTypes auf Behältertypen zugreifen

Die Methode FindContinerTypes gibt die Behältertypen eines angegebenen Moduls zurück. Das Modul wird über den Parameter typeIdentifier angegeben. Die resultierende Liste enthält die TypeIdentifiers sämtlicher Behältertypen des angeforderten Typs. Üblicherweise umfasst dies ein Gerät und einen Baugruppenträger, und die Behälter werden in ihrer Reihenfolge in der Hierarchie im Projekt mit Beginn am Gerät angegebenen.

Name des Parameters	Typ	Beschreibung
`typeIdentifier`	String	Typkennung eines Gerätelements

ACHTUNG
Diese Methode funktioniert nur bei die in der Netzsicht sichtbaren Modulen.
Diese Methode funktioniert nur bei Modulen, nicht bei Submodulen.

```
string typeIdentifier = ...;
string[] containerTypes = moduleInformationProvider.FindContainerTypes(typeIdentifier);
```

Programmcode: Mit der Methode FindModuleTypes auf Versionen zugreifen

Die Methode FindModuleTypes gibt alle möglichen Versionen eines Hardwareobjekts unter Verwendung der teilweisen Typkennung des Geräteelements zurück. Diese Methode gibt eine Liste von Strings aus. Jeder String entspricht dem vollständigen TypeIdentifier einer möglichen Übereinstimmung für den teilweisen TypeIdentifier.

Eine gültige teilweise Typkennung darf nur vollständige Teile enthalten, wobei jeder Teil in der Typkennung durch das Zeichen "/" getrennt ist. Platzhalter oder unvollständige Teile werden nicht unterstützt. Außerdem müssen die folgenden Einschränkungen im Hinblick auf die Mindestanzahl von angegebenen Teilen beachten werden:

- Bestellnummer mindestens ein Teil. Beispiel: `OrderNumber:6ES7 317-2EK14-0AB0`
- GSD: mindestens zwei Teile. Beispiel: `GSD:SI05816A.GSD/M`
- System: mindestens ein Teil. Beispiel: `System:Rack.ET200SP`

Name des Parameters	Typ	Beschreibung
`partialTypeIdentifier`	String	teilweise Typkennung eines Geräteelements

```
string partialTypeIdentifier = ...;
string[] moduleTypes = moduleInformationProvider.FindModuleTypes(partialTypeIdentifier);
```

Programmcode: Mit der Methode GetPlugLocations auf die Steckorte zugreifen

Die Methode GetPlugLocations gibt die Informationen über Steckplätze wie Steckort, Positionsnummer (Bezeichnung eines Steckplatzes) und verfügbare Steckplätze für das Hardwareobjekt zurück.

Die Klasse PlugLocation hat die folgenden Eigenschaften.

Name der Eigenschaft	Typ	Beschreibung
`PositionNumber`	Int	Die Positionsnummer des freien Steckplatzes
`Label`	String	Die Bezeichnung des freien Steckplatzes

- Ist für eine bestimmte Positionsnummer kein "label" vorhanden, wird die String-Darstellung der Positionsnummer verwendet.
- `PlugLocation`-Objekte werden nur für freie Steckplätze bereitgestellt.

```
IHardwareObject hardwareObject = ...;
IList<PlugLocation> result = hardwareObject.GetPlugLocations();
foreach (PlugLocation item in result)
 {
    Console.WriteLine("{0} - {1}", item.PositionNumber, item.Label);
 }
```

7.16.3 Geräteelemente in einen anderen Steckplatz verschieben

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Mit der Aktion PlugMove(DeviceItem deviceItem, int positionNumber) von HardwareObject kann ein vorhandenes Geräteelement verschoben und in ein vorhandenes Hardwareobjekt gesteckt werden. Die Methode PlugMove fügt die Geräteelemente ein, wo das Modul die Bedienoberfläche nicht stecken konnte. In diesen Fällen wird die Aktion PlugMove mit Übersetzungsfehlern beendet.

Die Aktion CanPlugMove(DeviceItem deviceItem, int positionNumber) dient zum Ermitteln der Bewegungsmöglichkeit. Wenn die Bewegung nicht möglich ist, gibt CanPlugMove den Wert False zurück. Wenn die Methode den Wert True zurückgibt, kann die Aktion aus den folgenden unvorhersehbaren Gründen dennoch fehlschlagen.

- eine Positionsnummer wird bereits von einem anderen Geräteelement verwendet
- das aktuelle Geräteelement kann nicht in die Position gesteckt werden, obwohl sie frei ist
- der Behälter stellt die Positionsnummer nicht bereit
- der Name des Geräteelements wird bereits von einem vorhandenen Geräteelement im selben Behälter verwendet

- das Geräteelement kann nicht in den Behälter gesteckt werden
- das Geräteelement kann vom Benutzer nicht gesteckt werden
- das Geräteelement kann vom Benutzer nicht entfernt werden
- das Gerät ist online

Programmcode

Ändern Sie folgenden Programmcode:

```
HardwareObject hwObject = ...;
DeviceItem deviceItemToMove = ...;
int positionNumber = ...;
if(hwObject.CanPlugMove(deviceItemToMove, positionNumber)
{
    DeviceItem movedDeviceItem = hwObject.PlugMove(deviceItemToMove, positionNumber);
}
```

7.16.4 Geräteelement kopieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Verwenden Sie die Aktion PlugCopy(DeviceItem deviceItem, int positionNumber) von HardwareObject, um ein Gerät innerhalb eines Projekts zu kopieren und in vorhandene Hardware zu stecken. In seltenen Fällen kann die Methode PlugCopy dann funktionieren, wenn eine Baugruppe nicht in die UI gesteckt werden kann. In diesem Fall treten nach dem Kopieren Übersetzungsfehler auf. Wenn PlugCopy erfolgreich war, wird die Kopie des Geräteelementobjekts zurückgegeben. Andernfalls wird eine wiederherstellbare Ausnahme ausgelöst.

Mögliche Ursachen für eine fehlgeschlagene Aktion:

- eine Positionsnummer wird bereits von einem anderen Geräteelement verwendet
- das aktuelle Geräteelement kann nicht in die Position gesteckt werden, obwohl sie frei ist
- der Behälter stellt die Positionsnummer nicht bereit
- der Name des Geräteelements wird bereits von einem vorhandenen Geräteelement im selben Behälter verwendet
- das Geräteelement kann nicht in den Behälter gesteckt werden

- das Geräteelement kann nicht in die UI gesteckt werden
- ...

Mit der Aktion CanPlugCopy(DeviceItem deviceItem, int positionNumber) können Sie bestimmen, ob der Kopiervorgang möglich ist. Wenn der Kopiervorgang nicht ausgeführt werden kann, gibt CanPlugCopy den Wert False zurück. Wenn die Methode jedoch den Wert True zurückgibt, kann die Aktion aus unvorhersehbaren Gründen dennoch fehlschlagen.

Name des Parameters	Typ	Beschreibung
deviceItem	DeviceItem	Zu kopierendes Geräteelement
positionNumber	Int	Positionsnummer zum Kopieren des Geräteelements

Programmcode

Ändern Sie folgenden Programmcode:

```
HardwareObject hwObject = ...;
DeviceItem deviceItemToCopy = ...;
int positionNumber = ...;
if(hwObject.CanPlugCopy(deviceItemToCopy, positionNumber))
{
   DeviceItem copiedDeviceItem = hwObject.PlugCopy(deviceItemToCopy, positionNumber);
}
```

7.16.5 Geräteelement löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um ein Geräteelement zu löschen, ändern Sie folgenden Programmcode:

```
Project project = ...;
var device = project.UngroupedDevicesGroup.Devices.Find("......");
var deviceItem = deviceItem.DeviceItems.First();

// delete device item
deviceItem.Delete();
```

7.16.6 Geräteelemente enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Um ein Geräteelement abzurufen, verwenden Sie das HardwareObject Die Elemente eines Hardwareobjekts sind die in das Hardwareobjekt gesteckten Teile, die der Anwender im TIA Portal sehen kann:

- ein Baugruppenträger in einem Gerät
- ein Modul in einem Baugruppenträger
- Ein Submodul in einem Modul
- Ein Submodul in einem Submodul

Hinweis

Weiterführende Informationen zu diesem Thema finden Sie im Kapitel Hierarchie von Hardware-Objekten des Objektmodells (Seite 64).

Programmcode: Geräteelement eines Geräts enumerieren

Ändern Sie den folgenden Programmcode, um Geräteelemente eines Hardware-Objekts zu enumerieren:

```
private static void EnumerateDeviceItems(HardwareObject hardwareObject)
{
    foreach (DeviceItem deviceItem in hardwareObject.Items)
    {
        // add code here
    }
}
```

Programmcode: Mit Zusammensetzungshierarchie enumerieren

Ändern Sie den folgenden Programmcode, wenn Sie die Geräteelemente eines Geräts mit Hilfe der Zusammensetzungshierarchie enumerieren möchten:

```
//Enumerates devices using an composition
private static void EnumerateDeviceItems(Device device)
{
    DeviceItemComposition deviceItemComposition = device.DeviceItems;
    foreach (DeviceItem deviceItem in deviceItemComposition)
    {
        // add code here
    }
}
```

Programmcode: Geräteelemente mit Zuordnung enumerieren

Ändern Sie den folgenden Programmcode, um die Geräteelemente mit einer Zuordnung zu enumerieren:

```
//Enumerates devices using an association
private static void EnumerateDeviceItemsWithAssociation(Device device)
{
    DeviceItemAssociation deviceItemAssociation = device.Items;
    foreach (DeviceItem deviceItem in deviceItemAssociation)
    {
        // add code here
    }
}
```

7.16.7 Geräteelemente aufrufen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung: Geräteelemente aufrufen

Zum Aufrufen von Objekten vom Typ "DeviceItem" verwenden Sie die folgenden Attribute:

- Name (string): Name des Geräteelements
- Behälter (HardwareObject): Behälter, in den das Geräteelement gesteckt wird

Name	Datentyp	Schreibbar	Zugriff	Beschreibung
Author	String	read/write	dynamisch	
Comment	String	read/write	dynamisch	
FirmwareVersion	String	read	dynamisch	Nur für Kopfmodule
GsdName	String	read	dynamisch	Name der GSD-Datei.
GsdType	String	read	dynamisch	Typ des Hardwareobjekts. Der Wert des Geräts ist stets "D".
GsdId	String	read	dynamisch	Spezifische Kennung des Hardwareobjekts. Für Geräte immer leer.
IsBuiltIn	Bool	read		
IsGsd	Bool	read		WAHR für GSD-Gerät oder GSDML-Gerät
IsPlugged	Bool	read		
IsProfibus	Bool	read		
IsProfinet	Bool	read		
Name	String	read/write		
OrderNumber	String	read	dynamisch	Nur für Kopfmodule
PositionNumber	Bool	read		
TypeIdentifier	String	read		

Programmcode: Geräteelement aufrufen

Ändern Sie zum Aufrufen eines Geräteelements den folgenden Programmcode:

```
public static DeviceItem AccessDeviceItemFromDevice(Device device)
{
    DeviceItem deviceItem = device.DeviceItems[0];
    return deviceItem;
}
```

Programmcode: Auf Geräteelement von einem Geräteelement zugreifen

Ändern Sie den folgenden Programmcode, um auf ein Geräteelement eines Geräteelements zuzugreifen:

```
public static DeviceItem AccessDeviceItemFromDeviceItem(DeviceItem deviceItem)
{
    DeviceItem subDeviceItem = deviceItem.DeviceItems[0];
    return subDeviceItem;
}
```

Programmcode: Zum Behälter eines Geräteelements navigieren

Ändern Sie den folgenden Programmcode über das Attribut "Container" von DeviceItem, um zurück zum Behälter eines Geräteelements zu navigieren:

```
DeviceItem deviceItem = ...;
HardwareObject container = deviceItem.Container;
```

Programmcode: Identifikationsattribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Device device = ...;
var attributeNames = new[] {
    "GsdName", "GsdType", "GsdId" };
foreach (var attributeName in attributeNames) {
    object attributeValue = ((IEngineeringObject)deviceItem).GetAttribute(attributeName);
    }
```

Programmcode: Attribute abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
GsdDeviceItem gsdDeviceItem =
((IEngineeringServiceProvider)deviceItem).GetService<GsdDeviceItem>();

string gsdName = gsdDeviceItem.GsdName;
string gsdType = gsdDeviceItem.GsdType;
string gsdId = gsdDeviceItem.GsdId;
bool isProfinet = gsdDeviceItem.IsProfinet;
bool isProfibus = gsdDeviceItem.IsProfibus;;
```

Programmcode: Attribute bei dynamischem Zugriff abrufen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
GsdDeviceItem gsdDeviceItem =
((IEngineeringServiceProvider)deviceItem).GetService<GsdDeviceItem>();

var attributeNames = new[] {
    "TypeName", "Author", "Comment", ...
    ;
foreach (var attributeName in attributeNames) {
    object attributeValue =
((IEngineeringObject)gsdDeviceItem).GetAttribute(attributeName);
    }
```

Programmcode: Attribute festlegen

Um die Attribute festzulegen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
((IEngineeringObject)deviceItem).SetAttribute("Comment", "This is a comment.");
```

Programmcode: prm-Daten eines Kopfmoduls abrufen

Um die prm-Daten abzurufen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
GsdDeviceItem gsdDeviceItem =
((IEngineeringServiceProvider)deviceItem).GetService<GsdDeviceItem>();

int dsNumber = 0;       // For Profibus GSDs, dataset number zero must be used!
int byteOffset = 0;
int lengthInBytes = 5;

// read complete data set:
byte[] prmDataComplete = gsdDeviceItem.GetPrmData(dsNumber, byteOffset, lengthInBytes);

// read partial data set (only second byte):
byteOffset = 1;
lengthInBytes = 1;
byte[] prmDataPartial = gsdDeviceItem.GetPrmData(dsNumber, byteOffset, lengthInBytes);
```

Programmcode: prm-Daten eines Kopfmoduls einstellen

Um die prm-Daten abzurufen, ändern Sie folgenden Programmcode:

```
DeviceItem deviceItem = ...;
GsdDeviceItem gsdDeviceItem =
((IEngineeringServiceProvider)deviceItem).GetService<GsdDeviceItem>();

// The parameters byteOffset and the length of the byte array prmData define the range
within the
// dataset which is written to.
// For Profibus GSDs, dataset number zero must be used!

// Change the highlighted bytes 2-4 from 0x0 to 0x1
// to write only the first two bytes: byte[] prmData = {0x05, 0x21};

int dsNumber = 0;
int byteOffset = 0;
byte[] prmData = {0x05, 0x21, 0x01, 0x01, 0x01};

gsdDeviceItem.SetPrmData(dsNumber, byteOffset, prmData);
```

Siehe auch

Hierarchie von Hardware-Objekten des Objektmodells (Seite 64)

7.16.8 Auf das Geräteelement als Schnittstelle zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Wenn ein Geräteelement eine Schnittstelle ist, bietet es gegenüber einem einfachen Geräteelement zusätzliche Funktionalität. Über diese Schnittstelle kann der Anwender auf die Teilnehmer und die Betriebsart der Schnittstelle zugreifen. Aufgrund dieser Funktionalität kann das Geräteelement durch Zugriff auf die Funktion NetworkInterface als IoDevice (Slave) oder als IoController (Master) verwendet werden (ein spezifischer Dienst des Geräteelements).

Auf die Eigenschaften der Schnittstelle wird über die Enumeration InterfaceOperatingModes zugegriffen.

Wert	Beschreibung
InterfaceOperatingModes.None	Standardeinstellung
InterfaceOperatingModes.IoDevice	Betriebsart der Schnittstelle "IoDevice" (Slave).
InterfaceOperatingModes.IoController	Betriebsart der Schnittstelle "IoController" (Master).
InterfaceOperatingModes.IoDevice or InterfaceOperatingModes.IoController	Schnittstellenbetrieb: beide der oben genannten.

Programmcode: Auf die Netzwerkschnittstellenfunktion zugreifen

Um die Netzwerkschnittstellenfunktion abzurufen, ändern Sie folgenden Programmcode:

```
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;

//Modififying the operating mode and interface type
itf.InterfaceOperatingMode = InterfaceOperatingModes.IoDevice;
itf.InterfaceType = NetType.Profibus

//Accessing the ports linked to an interface.
NetworkPortAssociation nodes = itf.Ports;
```

7.16.9 Auf Attribute einer I/O-Geräteschnittstelle zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Für Schreibzugriff ist der PLC offline.

Verwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Attribute für IRT und isochronen Modus an der I/O-Geräteschnittstelle abrufen oder festlegen.

Zugriff auf die Schnittstelle eines I/O-Controllers

Über die folgenden Auf die folgenden Attribute kann auf die Schnittstelle eines I/O-Controllers zugegriffen werden. Der Controller muss der Synchronisierungsmaster sein:

Attributname	Datentyp	Schreibbar	Zugriff	Beschreibung
PnSendClock	Int64	r/w	Dynamisches Attribut	Takt in Nanosekunden senden

Zugriff auf die Schnittstelle eines I/O-Systems

Über die folgenden Attribute kann auf die Schnittstelle eines I/O-Systems zugegriffen werden. Die Ti/To-Werte können von allen Modulen und Submodulen, die zum I/O-System gehören, verwendet werden.

Attributname	Datentyp	Schreibbar	Zugriff
`IsochronousTiToAutoCalculation`	BOOL	r/w	Dynamisches Attribut
`IsochronousTi`	DOUBLE	r/w	Dynamisches Attribut
`IsochronousTo`	DOUBLE	r/w	Dynamisches Attribut

Zugriff auf die Schnittstelle eines I/O-Geräts

Über die folgenden Attribute kann auf die Schnittstelle eines I/O-Geräts zugegriffen werden. Die Ti/To-Werte können von allen Modulen und Submodulen, die zum I/O-System gehören, verwendet werden.

Attributname	Datentyp	Schreibbar	Zugriff
`IsochronousMode`	BOOL	r/w	Dynamisches Attribut
`IsochronousTiToCalculationMode`	IsochronousTiToCalculationMode	r/w	Dynamisches Attribut
`IsochronousTi`	DOUBLE	r/w	Dynamisches Attribut
`ochronousTo`	DOUBLE	r/w	Dynamisches Attribut

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

Wert	Beschreibung
`IsochronousTiToCalculationMode.None`	
`IsochronousTiToCalculationMode.FromOB`	Ti/To-Werte des OB (am IO-System konfiguriert) werden verwendet.
`IsochronousTiToCalculationMode.FromSubnet`	Dieser Wert wird von PROFINET-Schnittstellen nicht verwendet.
`IsochronousTiToCalculationMode.AutomaticMinimum`	Ti/To-Werte werden für das IO-Device automatisch berechnet.
`IsochronousTiToCalculationMode.Manual`	Der Anwender kann für dieses IO-Device Ti/To-Werte manuell eingeben.

Programmcode: Attribute einer I/O-Geräteschnittstelle abrufen oder festlegen

Ändern Sie den folgenden Programmcode, um auf den Sendetaktwert zuzugreifen:

```
DeviceItem pnInterface = ...;
// read attribute
long attributeValue = (long)pnInterface.GetAttribute("PnSendClock");
// write attribute
long sendClock = 2000000;
pnInterface.SetAttribute("PnSendClock", sendClock);
```

Ändern Sie den folgenden Programmcode, um auf die Ti/To-Werte eines OB zuzugreifen:

```
IoSystem ioSystem = ...;
bool titoAutoCalculation = (bool)ioSystem.GetAttribute("IsochronousTiToAutoCalculation");
ioSystem.SetAttribute("IsochronousTiToAutoCalculation", true);
```

Ändern Sie den folgenden Programmcode, um auf die isochrone Einstellung einer I/O-Geräteschnittstelle zuzugreifen:

```
DeviceItem pnInterface = ...;
bool isochronousMode = (bool)pnInterface.GetAttribute("IsochronousMode");
pnInterface.SetAttribute("IsochronousMode", true);
```

7.16.10 Auf Attribute des IoController zugreifen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Für Schreibzugriff ist der PLC offline.

Anwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Attribute für den IoController abrufen oder festlegen. Die folgenden Attribute sind nur unter PROFINET IoController (unter einer PROFINET-Schnittstelle) verfügbar. Wenn der Anwender ein Attribut in der Benutzeroberfläche ändern kann, kann der Anwender das entsprechende Attribut auch über TIA Portal Openness ändern.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
SyncRole	SyncRole	Lesen-Schreiben	Dynamisches Attribut	
PnDeviceNumber	Int	Schreibgeschützt	Dynamisches Attribut	In der TIA Portal UI befindet sich diese Eigenschaft unter dem Knoten Ethernet (Abschnitt PROFINET)

Die Eigenschaft Synchronication role ist in der PROFINET-Schnittstelle der TIA Portal UI verfügbar. Die Enum SyncRole hat die folgenden Werte:

Enumerationswert	Numerischer Wert
SyncRole.NotSynchronized	0
SyncRole.SyncMaster	1
SyncRole.SyncSlave	2
SyncRole.RedundantSyncMaster	4

Programmcode: Attribute des IoController festlegen

```
IoController ioController= ...;
SyncRole syncRole = (SyncRole)((IEngineeringObject)ioController).GetAttribute("SyncRole");
((IEngineeringObject)ioController).SetAttribute("SyncRole", SyncRole.SyncMaster);
```

7.16.11 Auf Attribute des IoConnector zugreifen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Für Schreibzugriff ist der PLC offline.

Anwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Attribute für den IoConnector abrufen oder festlegen. Die folgenden Attribute sind nur unter PROFINET IoController (unter einer PROFINET-Schnittstelle) verfügbar. Wenn der Anwender ein Attribut in der Benutzeroberfläche ändern kann, kann der Anwender das entsprechende Attribut auch über TIA Portal Oopenness ändern.

Es gibt vier Typen von Attributen wie Attribute für die Aktualisierungszeit, Attribute für die Überwachungszeit, Attribute für die Synchronisation und Attribute für die Gerätenummer.

Attribute für die Aktualisierungszeit

Die Attribute für die Aktualisierungszeit werden nachstehend aufgeführt.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
PnUpdateTimeAuto-Calculation	Bool	Lesen-Schreiben	Dynamisches Attribut	Wenn dieses Attribut wahr ist, wird die Aktualisierungszeit automatisch berechnet.
PnUpdateTime	Int64	Lesen-Schreiben	Dynamisches Attribut	Die Aktualisierungszeit wird in Nanosekunden gemessen.
PnUpdateTimeAdaption	Bool	Lesen-Schreiben	Dynamisches Attribut	

Attribute für die Überwachungszeit

Die Attribute für die Überwachungszeit werden nachstehend aufgeführt.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
PnWatchdogFactor	Int32	Lesen-Schreiben	Dynamisches Attribut	
PnWatchdogTime	Int64	Schreibgeschützt	Dynamisches Attribut	Die Überwachungszeit wird in Nanosekunden gemessen.

Attribute für die Synchronisation

Die Attribute für die Synchronisation werden nachstehend aufgeführt.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
RtClass	RtClass	Lesen-Schreiben	Dynamisches Attribut	
SyncRole	SyncRole	Schreibgeschützt	Dynamisches Attribut	

Die Enum RtClass hat folgende Werte.

Enumerationswert	Numerischer Wert
RtClass.None	0
RtClass.RT	1
RtClass.IRT	2

Die Enum SyncRole hat folgende Werte.

Enumerationswert	Numerischer Wert
SyncRole.NotSynchronized	0
SyncRole.SyncMaster	1
SyncRole.SyncSlave	2
SyncRole.RedundantSyncMaster	4

Attribute für die Gerätenummer

Die Attribute für die Gerätenummer werden nachstehend aufgeführt.

Attributname	Datentyp	Typ	Zugriff	Beschreibung
PnDeviceNumber	Int	Lesen-Schreiben	Dynamisches Attribut	Gibt die Gerätenummer an.

Programmcode: Attribute des IoConnector abrufen und festlegen

```
IoConnector connector = ...

var attributeNames = new[] {
"PnUpdateTimeAutoCalculation", "PnUpdateTime", "PnUpdateTimeAdaption", "PnWatchdogFactor",
"PnWatchdogTime", "RtClass", "SyncRole"
};

foreach (var attributeName in attributeNames)
 {
      object attributeValue = ((IEngineeringObject)connector ).GetAttribute(attributeName);
 }

connector.SetAttribute("PnUpdateTimeAutoCalculation", true);
```

Siehe auch

Projekt öffnen (Seite 99)

7.16.12 Auf einen Adresscontroller zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Ein Geräteelement, das zugleich ein Adresscontroller ist, bietet zusätzliche Funktionalität. Um auf die registrierten Adressen des Adresscontrollers zuzugreifen, muss die Rolle AddressController verwendet werden.

Programmcode: Adresscontroller abrufen

Um die Adresscontrollerrolle abzurufen, ändern Sie folgenden Programmcode:

```
AddressController addressController =
((IEngineeringServiceProvider)deviceItem).GetService<AddressController>();
    if (addressController != null)
        {
            ... // work with the address controller
        }
```

Attribute eines Adresscontrollers

Ein Adresscontroller hat folgende Attribute:

- RegisteredAddresses

Um die Attribute eines Adresscontrollers abzurufen, ändern Sie folgenden Programmcode:

```
AddressController addressController = ...;
foreach (Address registeredAddress in addressController.RegisteredAddresses)
        {
            ...
        }
```

7.16.13 Auf Adressen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Adressobjekte werden über den Zusammensetzungslink `Addresses` eines Geräteelements aufgerufen. Das Attribut `Addresses` gibt eine Sammlung von `AddressComposition` zurück, die enumeriert werden kann.

Programmcode: Adresse eines Geräteelements abrufen

Ändern Sie den folgenden Programmcode, um die Adresse eines Geräteelements abzurufen:

```
AddressComposition addresses = deviceItem.Addresses;
    foreach(Address address in addresses)
        {
            // work with the address
        }
```

Programmcode: Adresse eines IO-Controllers abrufen

Ändern Sie den folgenden Programmcode, um die Adresse eines IO-Controllers abzurufen:

```
AddressComposition addresses = ioController.Addresses;
    foreach(Address address in addresses)
        {
            // work with the address
        }
```

Attribute

Adresse unterstützt die folgenden Attribute:

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
AddressControl-lers	AddressControllerAsso-ciation	read		
Context	enum: AddressContext	read	dynamisch	nur für Diagnoseadressen und für spezifische Geräteelemente
IoType	enum: AddressIoType	read		
StartAdress	Int32	read/write		
Length	Int32	read		

Wert	Beschreibung
AddressIoType.Diagnosis	Der Adress-IO-Typ ist Diagnose.
AddressIoType.Input	Der Adress-IO-Typ ist Eingang.
AddressIoType.Output	Der Adress-IO-Typ ist Ausgang.
AddressIoType.Substitute	Der Adress-IO-Typ ist Ersatz.
AddressIoType.None	Der Adress-IO-Typ ist nicht angegeben.

Wert	Beschreibung
AddressContext.None	Der Adresskontext ist nicht gültig.
AddressContext.Device	Ein Geräte-Adressenkontext
AddressContext.Head	Ein Kopf-Adressenkontext

Programmcode: Attribute lesen

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
AddressControllerAssociation addressControllers = address.AddressControllers;
Int32 startAddress = address.StartAddress;
AddressIoType addressType = address.IoType;
Int32 adressLength = address.Length;
```

Programmcode: Attribute schreiben

Um die Attribute zu schreiben, ändern Sie folgenden Programmcode:

```
Address addressControllers = ...;

address.StartAddress = intValueStartAddress;
```

Programmcode: Attribute bei dynamischem Zugriff

Um die Attribute abzurufen, ändern Sie folgenden Programmcode:

```
Address address= ...;

object attributeValue = ((IEngineeringObject)address).GetAttribute("Context");
```

7.16.14 Auf "Hardwarekennung" zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Hardwarekennungsobjekte werden von den folgenden Objekten abgerufen:

- Device
- DeviceItem
- IoSystem

Die Hardwarekennung wird durch die Klasse `HwIdentifier` dargestellt und über das Attribut`HwIdentifiers`. abgerufen.

Programmcode: Hardwarekennung abrufen

Um HwIdentifier verfügbar zu machen, ändern Sie folgenden Programmcode:

```
var hwObject = ...
    foreach(HwIdentifier hardwareIdentifier in hwObject.HwIdentifiers)
       {
          // Work with the HwIdentifer
       }
```

Attribute einer Hardwarekennung

```
HwIdentifierControllerAssociation controllers = hwIdentifier.HwIdentifierControllers;
Int64 Identifier = hwIdentifier.Identifier;
```

7.16.15 Auf Hardwarekennungscontroller zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Wenn ein Geräteelement gleichzeitig ein Hardwarekennungscontroller ist, kann auf die registrierten Hardwarekennungen zugegriffen werden. Um auf diese HwIdentifierController zuzugreifen, muss ein bestimmter Dienst des Geräteelements verwendet werden.

Programmcode: Hardwarekennungscontroller abrufen

Um den HwIdentifierController abzurufen, ändern Sie folgenden Programmcode:

```
HwIdentifierController hwIdentifierController =
((IEngineeringServiceProvider)deviceItem).GetService<HwIdentifierController>();
    if (hwIdentifierController != null)
        {
            ... // work with the hardware identifier controller
        }
```

Programmcode: Attribute eines Hardwarekennungscontrollers

Ein Adresscontroller hat folgende Attribute:

- `RegisteredHwIdentifiers`: Die Hardwarekennungscontroller, auf denen die Hardwarekennung registriert ist.

Um die Attribute eines Adresscontrollers abzurufen, ändern Sie folgenden Programmcode:

```
HwIdentifierController hwIdentifierController = ...;
HwIdentifierAssociation controllers = hwIdentifierController.RegisteredHwIdentifiers;
```

7.16.16 Auf Kanäle von Geräteelementen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Ein Kanal wird durch die Klasse Channel dargestellt. Kanäle werden von einem Geräteelement über das Attribut Channels der Klasse DeviceItem abgerufen. Das Attribut Channels gibt eine Implementierung von ChannelComposition zurück, die enumeriert werden kann. Wenn das Geräteelement keine Kanäle hat, gibt das Attribut Channels eine leere Sammlung zurück.

Obligatorische Attribute

Ein Kanal unterstützt die folgenden obligatorischen Attribute:

Attributname	Datentyp	Schreibbar	Zugriff	Kommentar
IoType	ChannelIoType	read		
Type	ChannelType	read		
Number	Int32	read		
ChannelAddress	Int32	read	dynamisch	Adresse des Kanals in Bits
ChannelWidth	UInt32	read	dynamisch	Breite des Kanals in Bits

Programmcode: Kanäle von einem Geräteelement abrufen

Ändern Sie den folgenden Programmcode, um die Kanäle eines Geräteelements abzurufen:

```
ChannelComposition channels = deviceItem.Channels
foreach(Channel channel in channels)
    {
        // work with the channel
    }
```

Programmcode: Obligatorische Attribute eines Kanals

Ändern Sie den folgenden Programmcode, um die Kanäle eines Geräteelements abzurufen:

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

Programmcode: Werte der Attribute bei dynamischem Zugriff abrufen

Um die Werte von dynamischen Attributen abzurufen, ändern Sie folgenden Programmcode:

```
Channel channel = ...;
Int32 channelAddress = (Int32)((IEngineeringObject)channel).GetAttribute("ChannelAddress");
UInt32 channelWidth = (UInt32)((IEngineeringObject)channel).GetAttribute("ChannelWidth");
```

Programmcode: Wert eines dynamischen Attributs festlegen

Um den Wert eines schreibbaren dynamischen Attributs festzulegen, ändern Sie folgenden Programmcode:

```
Channel channel = ...;
((IEngineeringObject)channel).SetAttribute("AnAttribute", 1234);
```

7.17 Funktionen auf Daten eines HMI-Gerätes

7.17.1 Bilder

7.17.1.1 Benutzerdefinierte Bilderordner erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um einen benutzerdefinierten Bilderordner zu erstellen, ändern Sie folgenden Programmcode:

```
//Creates a screen folder
private static void CreateScreenFolder(HmiTarget hmitarget)
{
    ScreenUserFolder myCreatedFolder =
hmitarget.ScreenFolder.Folders.Create("myScreenFolder");
}
```

7.17.1.2 Bild aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Hinweis

Sie können ein Permanentfenster nicht löschen. Ein Permanentfenster ist ein Systembild, das immer vorhanden ist.

Programmcode

Um ein Bild aus einem bestimmten Ordner zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteScreenFromFolder(HmiTarget hmiTarget)
{
    ScreenUserFolder screenUserFolder =
hmiTarget.ScreenFolder.Folders.Find("myScreenFolder");
    ScreenComposition screens = screenUserFolder.Screens;
    Screen screen = screens.Find("myScreenName");
    if (screen != null)
    {
        screen.Delete();
    }
}
```

7.17.1.3 Bildvorlage aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um eine Bildvorlage aus einem bestimmten Ordner zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteScreenTemplateFromFolder(HmiTarget hmiTarget)
{
    string templateName = "MyScreenTemplate";
    ScreenTemplateUserFolder folder =
hmiTarget.ScreenTemplateFolder.Folders.Find("myScreenTemplateFolder");
    ScreenTemplateComposition templates = folder.ScreenTemplates;
    ScreenTemplate template = templates.Find(templateName);
    if (template != null)
    {
        template.Delete();
    }
}
```

7.17.1.4 Alle Bilder aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Hinweis

Sie können ein Permanentfenster nicht löschen. Ein Permanentfenster ist ein Systembild, das immer vorhanden ist.

Programmcode

Um alle Bilder aus einem bestimmten Ordner zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteAllScreensFromFolder(HmiTarget hmitarget)
//Deletes all screens from a user folder or a system folder
{
    ScreenUserFolder folder = hmitarget.ScreenFolder.Folders.Find("myScreenFolder");
    //or ScreenSystemFolder folder = hmitarget.ScreenFolder;
    ScreenComposition screens = folder.Screens;
    List<Screen> list = new List<Screen>();
    foreach(Screen screen in screens)
    {
        list.Add(screen);
    }
    foreach (Screen screen in list)
    {
        screen.Delete();
    }
}
```

7.17.2 Zyklen

7.17.2.1 Zyklus löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Im Projekt ist ein HMI-Gerät vorhanden.

Verwendung

Standardzyklen können Sie nicht löschen.

Sie können anhand der Zusammensetzung im Objektmodell (composition count) des jeweiligen Zyklus ermitteln, ob tatsächlich Zyklen gelöscht wurden. Auf diese Zyklen ist kein Zugriff mehr möglich.

Programmcode

Um einen Zyklus aus einem HMI-Gerät zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteCycle(HmiTarget hmiTarget)
{
    CycleComposition cycles = hmiTarget.Cycles;
    Cycle cycle = cycles.Find("myCycle");
    cycle.Delete();
}
```

7.17.3 Textlisten

7.17.3.1 Textliste löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um eine ausgewählte Textliste und alle zugehörigen Listeneinträge aus einem HMI-Gerät zu löschen, ändern Sie folgenden Programmcode:

```
public static void DeleteTextList(HmiTarget hmiTarget)
{
    TextListComposition textLists = hmiTarget.TextLists;
    TextList textList = textLists.Find("myTextList");
    textList.Delete();
}
```

7.17.4 Grafiklisten

7.17.4.1 Grafikliste löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um eine ausgewählte Grafikliste und alle zugehörigen Listeneinträge aus einem HMI-Gerät zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteGraphicList(HmiTarget hmiTarget)
{
    GraphicListComposition graphicLists = hmiTarget.GraphicLists;
    GraphicList graphicList = graphicLists.Find("myGraphicList");
    graphicList.Delete();
}
```

7.17.5 Verbindungen

7.17.5.1 Verbindung löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um eine ausgewählte Kommunikationsverbindung aus einem HMI-Gerät zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteConnection(HmiTarget hmiTarget)
{
    ConnectionComposition connections = hmiTarget.Connections;
    Connection connection = connections.Find("HMI_connection_1");
    connection.Delete();
}
```

7.17.6 Variablentabelle

7.17.6.1 Benutzerdefinierte Ordner für HMI-Variablen erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um einen benutzerdefinierten Ordner für HMI-Variablen zu erstellen, ändern Sie folgenden Programmcode:

```
private static void CreateUserfolderForHMITags(HmiTarget hmitarget)
// Creates an HMI tag user folder
{
    TagSystemFolder folder = hmitarget.TagFolder;
    TagUserFolder myCreatedFolder = folder.Folders.Create("MySubFolder");
}
```

7.17.6.2 Variablen einer HMI-Variablentabelle enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um alle Variablen einer HMI-Variablentabelle zu enumerieren, ändern Sie folgenden Programmcode:

```
private static void EnumerateTagsInTagtable(HmiTarget hmitarget)
// //Enumerates all tags of a tag table
{
    TagTable table = hmitarget.TagFolder.TagTables.Find("MyTagtable");
    // Alternatively, you can access the default tag table:
    // TagTable defaulttable = hmitarget.TagFolder.DefaultTagTable;

    TagComposition tagComposition = table.Tags;
    foreach (Tag tag in tagComposition)
    {
        // Add your code here
    }
}
```

7.17.6.3 Einzelne Variable aus einer HMI-Variablentabelle löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um eine bestimmte Variable aus einer HMI-Variablentabelle zu löschen, ändern Sie folgenden Programmcode:

```
private static void DeleteATag(HmiTarget hmiTarget)
{
    string tagName = "MyTag";
    TagTable defaultTagTable = hmiTarget.TagFolder.DefaultTagTable;
    TagComposition Variablen = defaultTagTable.Tags;
    Tag tag = tags.Find(tagName);
    tag.Delete();
}
```

7.17.6.4 Variablentabelle aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Im Projekt ist ein HMI-Gerät vorhanden.

Verwendung

Sie können die Standardvariablentabelle nicht löschen

Programmcode

Ändern Sie folgenden Programmcode:

```
// Delete a tag table from a specific folder
private static void DeleteTagTable(HmiTarget hmiTarget)
{
    string tableName = "myTagTable";
    TagSystemFolder tagSystemFolder = hmiTarget.TagFolder;
    TagTableComposition tagTables = tagSystemFolder.TagTables;
    TagTable tagTable = tagTables.Find(tableName);
    tagTable.Delete();
}
```

7.17.7 VB-Skripte

7.17.7.1 Benutzerdefinierte Ordner für Skripte erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Um einen benutzerdefinierten Unterordner für Skripte in einem Systemordner oder einem anderen benutzerdefinierten Ordner zu erstellen, ändern Sie folgenden Programmcode:

```
private static void CreateFolderInScriptfolder(HmiTarget hmitarget)
//Creates a script user subfolderVBScriptSystemFolder
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptUserFolderComposition vbScriptFolders = vbScriptFolder.Folders;
    VBScriptUserFolder vbScriptSubFolder = vbScriptFolder.Folders.Create("mySubfolder");
}
```

7.17.7.2 VB-Skript aus einem Ordner löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Im Projekt ist ein HMI-Gerät vorhanden.

Programmcode

Um ein VB-Skript aus einem bestimmten Ordner zu löschen, ändern Sie folgenden Programmcode:

```
//Deletes a vbscript from a script folderVBScriptSystemFolder
private static void DeleteVBScriptFromScriptFolder(HmiTarget hmitarget)
{
    VBScriptUserFolder vbscriptfolder =
hmitarget.VBScriptFolder.Folders.Find("MyScriptFolder");
    var vbScripts = vbscriptfolder.VBScripts;
    if (null != vbScripts)
    {
        var vbScript = vbScripts.Find("MyScript");
        vbScript.Delete();
    }
}
```

7.17.8 Benutzerdefinierten Ordner eines Bediengeräts löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Ändern Sie den folgenden Programmcode, um einen benutzerdefinierten Ordner eines Bediengeräts zu löschen:

```
HmiTarget hmiTarget = ...;
ScreenUserFolder screenUserGroup = hmiTarget.ScreenFolder.Folders.Find("MyUserFolder");
screenUserGroup.Delete();
```

7.18 Funktionen auf Daten eines PLC-Gerätes

7.18.1 Status einer PLC ermitteln

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden. Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet. Siehe Projekt öffnen (Seite 99)

Verwendung

Sie können den Zustand einer PLC oder aller PLCs in einem Projekt ermitteln.

TIA Portal Openness unterscheidet zwischen folgenden Zuständen:

- Offline
- PLC ist verbunden („Verbindung wird hergestellt")
- Online
- PLC ist nicht verbunden („Verbindung wird getrennt")
- Inkompatibel
- Zugriff nicht möglich
- Geschützt

Programmcode

Ändern Sie folgenden Programmcode, um den Zustand eines PLC zu ermitteln:

```
public static OnlineState GetOnlineState(DeviceItem deviceItem)
{
    OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
    return onlineProvider.State;
}
```

Ändern Sie folgenden Programmcode, um den Zustand aller PLCs in einem Projekt zu ermitteln:

```
public static void DetermineOnlineStateOfAllProjectDevices(Project project)
{
    foreach (Device device in project.Devices)
    {
        foreach (DeviceItem deviceItem in device.DeviceItems)
        {
            OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
            if (onlineProvider != null)
            {
                OnlineState state = onlineProvider.State;
            }
        }
    }
}
```

7.18.2 Auf Parameter einer Online-Verbindung zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Mit der Schnittstelle TIA Portal Openness API können Sie Parameter für eine Online-Verbindung ermitteln oder festlegen:

- Verfügbare Verbindungsarten mit einer PLC enumerieren
- Verfügbare Schnittstellen zu einer PLC enumerieren
- Zugeordnete Steckplätze enumerieren
- Verfügbare Adressen der Subnetze und Gateways enumerieren
- Verbindungsparameter festlegen

Programmcode: Verbindungsparameter ermitteln

Um die verfügbaren Verbindungsarten, PC-Schnittstellen und Steckplätze zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumerateConnectionModesOfPLC(DeviceItem deviceItem)
{
    OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
    if (onlineProvider == null)
    {
       return; // Only cpu device items can provide OnlineProvider service
    }
    // Accessing connection configuration object
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    // Now access connection configuration members
    foreach (ConfigurationMode mode in configuration.Modes)
    {
        Console.WriteLine("Mode name:{0}", mode.Name);
        foreach (ConfigurationPcInterface pcInterface in mode.PcInterfaces)
        {
            Console.WriteLine("PcInterface name:{0}", pcInterface.Name);
            Console.WriteLine("PcInterface number:{0}", pcInterface.Number);
            foreach (ConfigurationTargetInterface targetInterface in
pcInterface.TargetInterfaces)
            {
                Console.WriteLine("TargetInterface:{0}", targetInterface.Name);
            }
        }
    }
}
```

Sie können auch über den Namen auf eine Verbindungsart und eine PC-Schnittstelle zugreifen:

```
public static ConfigurationTargetInterface
GetTargetInterfaceForOnlineConnection(OnlineProvider onlineProvider)
{
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    ConfigurationMode mode = configuration.Modes.Find("PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("PLCSIM", 1);
    ConfigurationTargetInterface slot = pcInterface.TargetInterfaces.Find("2 X3");
    return slot;
}
```

Um die an einer PC-Schnittstelle verfügbaren Adressen von Subnetzen und Gateways zu enumerieren, ändern Sie folgenden Programmcode:

```
public static void EnumeratingPCInterfaceSubnetsAndGateways(ConfigurationPcInterface
pcInterface)
{
    foreach (ConfigurationSubnet subnet in pcInterface.Subnets)
    {
        Console.WriteLine("Subnet name:{0}", subnet.Name);
        foreach (ConfigurationGateway gateway in subnet.Gateways)
        {
            //Get the name of the gateway:
            Console.WriteLine("Gateway name:{0}", gateway.Name);
            //Get the IP address of each gateway:
            foreach (ConfigurationAddress gatewayAddress in gateway.Addresses)
            {
                Console.WriteLine("Gateway Address:{0}", gatewayAddress.Name);
            }
        }
    }
}
```

Sie können auch über den Namen oder die IP-Adresse auf Subnetze und Gateways zugreifen:

```
public static void AccessSubnetAndGatewayOfPCInterface(ConfigurationPcInterface
pcInterface)
{
    ConfigurationSubnet subnet = pcInterface.Subnets.Find("PN/IE_1");
    ConfigurationAddress subnetAddress = subnet.Addresses.Find("192.168.0.1");
    ConfigurationGateway gateway = subnet.Gateways.Find("Gateway 1");
    ConfigurationAddress gatewayAddress = gateway.Addresses.Find("192.168.0.2");
}
```

Programmcode: Verbindungsparameter festlegen

Hinweis

Durch Festlegen der Verbindungsparameter werden alle zuvor festgelegten Verbindungsparameter überschrieben. Wenn Sie die Verbindungsparameter bereits direkt im TIA Portal festgelegt haben, brauchen Sie `ApplyConfiguration` nicht aufzurufen. Besteht bereits eine Online-Verbindung mit einem PLC, während `ApplyConfiguration` aufgerufen wird, wird eine Ausnahme ausgelöst.

Ändern Sie den folgenden Programmcode, um Steckplatzparameter festzulegen:

```
public static void SetConnectionWithSlot(OnlineProvider onlineProvider)
{
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    ConfigurationMode mode = configuration.Modes.Find(@"PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("PLCSIM", 1);
    // or network pc interface that is connected to plc
    ConfigurationTargetInterface slot = pcInterface.TargetInterfaces.Find("2 X3");
    configuration.ApplyConfiguration(slot);
    // After applying configuration, you can go online
    onlineProvider.GoOnline();
}
```

Ändern Sie den folgenden Programmcode, um Gateway-Adressparameter einzustellen:

```
public static void SetConnectionWithGatewayAddress(OnlineProvider onlineProvider, string
subnetName, string gatewayAddressName)
{
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    ConfigurationMode mode = configuration.Modes.Find(@"PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("PLCSIM", 1);
    // or network pc interface that is connected to plc
    ConfigurationSubnet subnet = pcInterface.Subnets.Find(subnetName);
    ConfigurationAddress gatewayAddress = subnet.Addresses.Find(gatewayAddressName);
    configuration.ApplyConfiguration(gatewayAddress);
    // After applying configuration, you can go online
    onlineProvider.GoOnline();
}
```

Ändern Sie den folgenden Programmcode, um Subnetz-Adressparameter einzustellen:

```
public static void SetConnectionWithSubnetAddress(OnlineProvider onlineProvider, string
subnetName)
{
    ConnectionConfiguration configuration = onlineProvider.Configuration;
    ConfigurationMode mode = configuration.Modes.Find(@"PN/IE");
    ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("PLCSIM", 1);
    // or network pc interface that is connected to plc
    ConfigurationSubnet subnet = pcInterface.Subnets.Find(subnetName);
    ConfigurationAddressComposition addresses = subnet.Addresses;
    configuration.ApplyConfiguration(addresses[0]);
    // After applying configuration, you can go online
    onlineProvider.GoOnline();
}
```

7.18.3 PLC von R/H-System online setzen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Anwendung

Mit dem Dienst RHonlineProvider können Sie entweder den primären PLC oder den Backup-PLC eines R/H-Systems online setzen.

Programmcode: Auf Dienst RHOnlineProvider von einem Gerät aus zugreifen

Ändern Sie folgenden Code, um auf RHOnlineProvider zuzugreifen:

```
Device device = project.Devices.Find("S7-1500R/H-System_1");
RHOnlineProvider rhOnlineProvider = device.GetService<RHOnlineProvider>();
```

Programmcode: Verbindungsparameter festlegen

Mit dem Objekt ConnectionConfiguration können Sie eine Verbindung zum Gerät herstellen. Der Zugriff ist möglich über die Eigenschaft Configuration von RHOnlineProvider. Weitere Informationen zur Herstellung der Verbindung finden Sie unter Auf Parameter einer Online-Verbindung zugreifen (Seite 264)

Ändern Sie den folgenden Programmcode, um über den Namen eine Verbindungsart einzurichten und auf eine PC-Schnittstelle zuzugreifen:

```
ConnectionConfiguration connectionConfiguration = rhOnlineProvider.Configuration;
ConfigurationMode mode = connectionConfiguration.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("Broadcom NetXtreme Gigabit
Ethernet", 1);
ConfigurationTargetInterface targetConfiguration = pcInterface.TargetInterfaces.Find("1
X1");
bool success = connectionConfiguration.ApplyConfiguration(targetConfiguration);
```

Hinweis

R/H-System besteht aus zwei PLCs, eine einzelne Verbindungskonfiguration wird Ihnen bereitgestellt.

Programmcode: R/H-System online setzen

Sie können entweder den primären PLC oder den Backup-PLC online setzen. Wenn versucht wird, beide Ziele gleichzeitig online zu setzen, gibt das System eine Ausnahme EngineeringTargetInvocationException zurück.

Ändern Sie folgenden Programmcode, um den primären PLC online zu setzen:

```
OnlineState onlineState = rhOnlineProvider.GoOnlineToPrimary();
```

Ändern Sie folgenden Programmcode, um den Backup-PLC online zu setzen:

```
OnlineState onlineState = rhOnlineProvider.GoOnlineToBackup();
```

Hinweis

Wenn Sie einen PLC eines R/H-Systems online setzen, dürfen Sie ein zuvor gespeichertes Passwort wiederverwenden.

Programmcode: Onlinestatus eines R/H-Systems feststellen

Sie können mit den Eigenschaften PrimaryState und BackupState von RHOnlineProvider den Online-Verbindungsstatus eines primären PLC und eines Backup-PLC getrennt voneinander feststellen . Beide Eigenschaften geben die Enumeration OnlineState zurück. Weitere Informationen dazu, wie Sie den Onlinezustand eines PLC ermitteln, finden Sie unter Zustand eines PLC feststellen (Seite 263)

Ändern Sie folgenden Programmcode, um den Zustand eines primären PLC und eines Backup-PLC zu ermitteln:

```
RHOnlineProvider rhOnlineProvider = ...;
OnlineState primaryState = rhOnlineProvider.PrimaryState;
OnlineState backupState = rhOnlineProvider.BackupState;
```

Programmcode: R/H-System offline setzen

Ändern Sie folgenden Programmcode, um ein R/H-System, das derzeit online ist, durch Aufruf der Methode RHOnlineProvider.GoOffline in den Offline-Zustand zu versetzen:

```
rhOnlineProvider.GoOffline();
```

Siehe auch

Auf Parameter einer Online-Verbindung zugreifen (Seite 264)

Status einer PLC ermitteln (Seite 263)

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

7.18.4 Auf Softwarebehälter über primären PLC eines R/H-Systems zugreifen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Anwendung

Für den Zugriff auf einen Softwarebehälter können Sie den primären PLC eines R/H-Systems verwenden. Beispielsweise stellt das R/H-System Softwarebehälter für einen primären PLC zur Verfügung, der als PLC_1 dargestellt wird. Wenn Sie jedoch versuchen, auf einen Softwarebehälter für einen Backup-PLC zuzugreifen, der als PLC_2 dargestellt wird, gibt das System null zurück.

Die Besonderheiten eines Softwarebehälters und seine Software-Eigenschaften werden unter Auf Software-Ziel zugreifen (Seite 113) beschrieben.

Programmcode: Auf Softwarebehälter zugreifen

Um auf Softwarebehälter über das primäre Gerät eines R/H-Systems zuzugreifen, ändern Sie den folgenden Programmcode:

```
foreach (DeviceItem deviceItem in rhDevice.DeviceItems)
{
  if (deviceItem.Name == "PLC_1")
  {
    SoftwareContainer softwareContainer = deviceItem.GetService<SoftwareContainer>();
    ... //Work with softwareContainer
  }
}
```

Siehe auch

Auf Software-Ziel zugreifen (Seite 113)

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

7.18.5 PLCs eines R/H-Systems laden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Anwendung

Sie können mit der TIA Portal Openness-Anwendung sowohl in den primären PLC als auch in den Backup-PLC eines R/H-Systems laden. Sie sollten die Möglichkeit haben, sowohl Hardware- als auch Softwarekomponenten des Systems zu laden. (Weitere Informationen finden Sie unter Hardware- und Softwarekomponenten ins PLC-Gerät laden (Seite 277))

Programmcode: RHDownloadProvider abrufen

Über RHDownloadProvider service können Sie aus einem Gerät in ein R/H-System laden.

Ändern Sie folgenden Programmcode, um RHDownloadProvider abzurufen:

```
...
Device device = project.Devices.Find("S7-1500R/H-System_1");
RHDownloadProvider rhDownloadProvider = device.GetService<RHDownloadProvider>();
...
```

Hinweis

Auf den Dienst DownloadProvider wird nicht im Zusammenhang mit CPUs zugegriffen, die zum R/H-System gehören.

Programmcode: IConfiguration abrufen

RHDownloadProvider stellt das Objekt ConnectionConfiguration über die Eigenschaft Configuration zur Verfügung, mit deren Hilfe die Verbindung zum Gerät konfiguriert wird.

Ändern Sie folgenden Programmcode, um das Objekt IConfiguration über ConnectionConfiguration an RHDownloadProvider abzurufen:

```
...
RHDownloadProvider rhDownloadProvider = device.GetService<RHDownloadProvider>();
ConnectionConfiguration connectionConfiguration = rhDownloadProvider.Configuration;
ConfigurationMode mode = connectionConfiguration.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("Broadcom NetXtreme Gigabit
Ethernet", 1);
IConfiguration targetConfiguration = pcInterface.TargetInterfaces.Find("1 X1");
...
```

Hinweis

R/H Systeme bestehen aus zwei PLCs. Es wird nur ein Verbindungskonfigurationsobjekt bereitgestellt, das zum Laden sowohl in primäre PLCs als auch in Backup-PLCs verwendet werden kann.

Programmcode: In primäre CPU und Backup-CPU laden

Ändern Sie folgenden Programmcode, um durch Aufrufen von RHDownloadProvider.DownloadToPrimary in den primären PLC zu laden:

```
DownloadResult DownloadToPrimary(IConfiguration
configuration,DownloadConfigurationDelegate
preDownloadConfigurationDelegate,DownloadConfigurationDelegate
postDownloadConfigurationDelegate,DownloadOptions downloadOptions);
```

Ändern Sie folgenden Programmcode, um durch Aufrufen von RHDownloadProvider.DownloadToBackup in den Backup-PLC zu laden:

```
DownloadResult DownloadToBackup(IConfiguration configuration,
DownloadConfigurationDelegate preDownloadConfigurationDelegate,
DownloadConfigurationDelegate postDownloadConfigurationDelegate, DownloadOptions
downloadOptions);
```

Parameter der Methode RHDownloadProvider

Sowohl RHDownloadProvider.DownloadToPrimary als auch RHDownloadProvider.DownloadToBackup akzeptieren dieselben Parameter und geben ebenfalls ein DownloadResult zurück. Weitere Informationen zu den Details von IConfiguration, DownloadConfigurationDelegate, DownloadOptions und DownloadResult finden Sie unter Hardware- und Softwarekomponenten ins PLC-Gerät laden (Seite 277)

Parametername	Typ	Beschreibung
configuration	Siemens.Engineering.Connection.IConfiguration	Konfiguration der Verbindung mit einem Gerät.
preDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Delegate, der aufgerufen wird, um die Konfiguration vor dem Laden zu prüfen
postDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Delegate, der aufgerufen wird, um die Konfiguration nach dem Laden zu prüfen
downloadOptions	Siemens.Engineering.Download.DownloadOptions	Download-Optionen

Sie haben je nach Zustand des R/H-Systems die Möglichkeit, über DownloadConfigurations für den Ladevorgang einen Stopp des Systems anzufordern. Deshalb werden zusätzlich zu der Konfiguration, die unter Hardware- und Softwarekomponenten ins PLC-Gerät laden

(Seite 277) beschrieben ist, auch die folgenden Datentypen zur Unterstützung von RHDownload hinzugefügt.

Konfiguration	Datentyp	Aktion	Beschreibung
DownloadSelection-Configuration	StopHSystem	Set CurrentSelection:StopH-SystemSelections. Verfügbare Enumerationswerte: • NoAction (Keine Aktion) • StopHSystem (R/H-System stoppen)	Die Module werden gestoppt, um das Laden in ein Gerät zu ermöglichen.
	StopHSystemOrModule	Set CurrentSelection:StopH-SystemOrModuleSelections. Verfügbare Enumerationswerte: • NoAction (Keine Aktion) • StopHSystem (R/H-System stoppen) • StopModule (Modul stoppen)	Die Module werden gestoppt, um das Laden in ein Gerät zu ermöglichen.
	StartBackupModules	Set CurrentSelection:StartBackupModulesSelections. Verfügbare Enumerationswerte: • NoAction (Keine Aktion) • SwitchToPrimaryCpu (Auf primär wechseln) • StartModule (Modul starten)	Module nach dem Laden ins Gerät starten.
	SwitchBackupToPrimary	Set CurrentSelection:Switch-BackupToPrimarySelections. Verfügbare Enumerationswerte: • NoAction (Keine Aktion) • SwitchToPrimaryCpu (Auf primär wechseln)	Module nach dem Laden ins Gerät starten.

Programmcode: Ladekonfiguration in Rückrufen handhaben

Ändern Sie folgenden Programmcode in Aufrufe von DownloadToPrimary und DownloadToBackup während der Behandlung von Konfigurationen in Rückrufen:

Beispiel für den Aufruf von Download

```
static void Main(string[] args)
{
 ...
 Project project = tiaPortal.Projects[0];
 Device device = project.Devices.Find("S7-1500R/H-System_1");
 RHDownloadProvider rhDownloadProvider = device.GetService<RHDownloadProvider>();
 ConnectionConfiguration connectionConfiguration = rhDownloadProvider.Configuration;
 ConfigurationMode mode = connectionConfiguration.Modes.Find("PN/IE");
 ConfigurationPcInterface pcInterface = mode.PcInterfaces.Find("Broadcom NetXtreme Gigabit
Ethernet", 1);
 IConfiguration targetConfiguration = pcInterface.TargetInterfaces.Find("1 X1");

 // Download to primary
 DownloadResult primaryDownloadResult =
rhDownloadProvider.DownloadToPrimary(targetConfiguration,
 PreConfigureDownloadCallback,
 PostConfigureDownloadCallback,
 DownloadOptions.Hardware | DownloadOptions.Software);
 WriteDownloadResults(primaryDownloadResult);
 // Download to backup
 DownloadResult backupDownloadResult =
rhDownloadProvider.DownloadToBackup(targetConfiguration,
 PreConfigureDownloadCallback,
 PostConfigureDownloadCallback,
 DownloadOptions.Hardware | DownloadOptions.Software);
 WriteDownloadResults(backupDownloadResult);
...
}
private static void PreConfigureDownloadCallback(DownloadConfiguration
downloadConfiguration)
{
  StopHSystem stopHSystem = downloadConfiguration as StopHSystem;
  if (stopHSystem != null)
     {
       stopHSystem.CurrentSelection = StopHSystemSelections.StopHSystem;
     }
  OverwriteTargetLanguages overwriteTargetLanguages = downloadConfiguration as
OverwriteTargetLanguages;
  if (overwriteTargetLanguages != null)
     {
       overwriteTargetLanguages.Checked = true;
     }
  AlarmTextLibrariesDownload alarmTextLibraries = downloadConfiguration as
AlarmTextLibrariesDownload;
  if (alarmTextLibraries != null)
     {
       alarmTextLibraries.CurrentSelection =
AlarmTextLibrariesDownloadSelections.ConsistentDownload;
       return;
     }
  CheckBeforeDownload checkBeforeDownload = downloadConfiguration as CheckBeforeDownload;
  if (checkBeforeDownload != null)
     {
       checkBeforeDownload.Checked = true;
       return;
```

Beispiel für den Aufruf von Download

```
      }
  ConsistentBlocksDownload consistentBlocksDownload = downloadConfiguration as
ConsistentBlocksDownload;
  if (consistentBlocksDownload != null)
      {
         consistentBlocksDownload.CurrentSelection =
ConsistentBlocksDownloadSelections.ConsistentDownload;
         return;
      }
  OverwriteSystemData overwriteSystenData = downloadConfiguration as OverwriteSystemData;
  if (overwriteSystenData != null)
      {
         overwriteSystenData.CurrentSelection = OverwriteSystemDataSelections.Overwrite;
         return;
      }
}
private static void PostConfigureDownloadCallback(DownloadConfiguration
downloadConfiguration)
{
   StartModules startModules = downloadConfiguration as StartModules;
   if (startModules != null)
   {
     startModules.CurrentSelection = StartModulesSelections.StartModule;
     return;
   }
}
private static void WriteDownloadResults(DownloadResult result)
{
   Console.WriteLine("State:" + result.State);
   Console.WriteLine("Warning Count:" + result.WarningCount);
   Console.WriteLine("Error Count:" + result.ErrorCount);
   RecursivelyWriteMessages(result.Messages);
}
private static void RecursivelyWriteMessages(DownloadResultMessageComposition messages,
string indent = "")
{
   indent += "\t";
   foreach (DownloadResultMessage message in messages)
   {
      Console.WriteLine(indent + "DateTime: " + message.DateTime);
      Console.WriteLine(indent + "State: " + message.State);
      Console.WriteLine(indent + "Message: " + message.Message);
      RecursivelyWriteMessages(message.Messages, indent);
   }
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

Hardware- und Softwarekomponenten ins PLC-Gerät laden (Seite 277)

7.18.6 Funktionen zum Laden von Daten ins PLC-Gerät

7.18.6.1 Hardware- und Softwarekomponenten ins PLC-Gerät laden

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Der Openness-Anwender kann über DownloadProvider (Zugriff über das DeviceItem) Software- und Hardwarekomponenten ins PLC-Gerät laden. Wenn ein DeviceItem ein ladbares Ziel darstellt, wird bei Aufruf von GetService eine Instanz von DownloadProvider zurückgegeben. Andernfalls gibt der Dienst null zurück.

Programmcode: DownloadProvider-Dienst aus einem Geräteelement abrufen

```
DeviceItem deviceItem = ...;
DownloadProvider downloadProvider = deviceItem.GetService<DownloadProvider>();
if (downloadProvider != null)
{   ...
}
```

Parameter der Download-Methode

Um Daten in ein PLC-Gerät zu laden, ruft der Anwender die Methode Download von DownloadProvider auf. Die Download-Methode hat vier Parameter, bei denen es sich um das Objekt IConfiguration, zwei Delegates und DownloadOptions (Hardware, Software oder Hardware und Software) handelt.

Parametername	Typ	Beschreibung
configuration	Siemens.Engineering.Connection.IConfiguration	Konfiguration der Verbindung mit einem Gerät.
preDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Aufzurufender Delegate zum Prüfen der Konfiguration vor dem Ladevorgang.
postDownloadConfigurationDelegate	Siemens.Engineering.Download.DownloadConfigurationDelegate	Aufzurufender Delegate zum Prüfen der Konfiguration nach dem Ladevorgang.
downloadOptions	Siemens.Engineering.Download.DownloadOptions	Ladeoptionen.

- Der Openness-Ladevorgang wird nur unterstützt, wenn die Konfigurationen vom Anwender sachgemäß gehandhabt werden. Wenn die Konfiguration ungültig ist, wird EngineeringTargetInvocationException ausgelöst und der Ladevorgang abgebrochen. Die F-aktivierten PLCs werden für den Ladevorgang nicht unterstützt.
- Da die Übersetzung ein Teil des Ladevorgangs ist, ist es empfehlenswert, die Übersetzung vor dem Ladevorgang durchzuführen, um die Übersetzungsergebnisse zu analysieren.
- Openness unterstützt nur die Option Gesamtladen.

Parameter 1: IConfiguration

Der Anwender muss das Objekt IConfiguration als ersten Parameter der Download-Methode angeben. Es dient dazu, eine Verbindung zum angegebenen PLC-Gerät herzustellen. Die Schnittstelle IConfiguration wird von ConfigurationAddress und ConfigurationTargetInterface implementiert. Auf beide Objekte kann über die Instanz ConnectionConfiguration zugegriffen werden. Die Instanz ConnectionConfiguration kann über die Eigenschaft DownloadProvider.Connection: ConnectionConfiguration oder optional über OnlineProvider.Connection: ConnectionConfiguration ermittelt werden.

Die Konfigurierung des Objekts ConnectionConfiguration wird im Abschnitt Auf Parameter einer Online-Verbindung zugreifen (Seite 264) beschrieben.

```
...
   DownloadProvider downloadProvider = null;
   ConnectionConfiguration configuration = downloadProvider.Configuration;
   ConfigurationMode configurationMode = configuration.Modes.Find("PN/IE");
   ConfigurationPcInterface pcInterface = configurationMode.PcInterfaces.Find("Intel(R)
Ethernet Connection I217-LM", 1);
   IConfiguration targetConfiguration = pcInterface.TargetInterfaces[0];
...
```

Parameter 2 und 3: DownloadConfigurationDelegate

Der Openness-Anwender muss zwei Implementierungen von leeren DownloadConfigurationDelegate(DownloadConfiguration downloadConfiguration) bereitstellen. Der erste Delegate wird für Konfigurationen vor dem Ladevorgang aufgerufen und der zweite wird nach Abschluss des Ladevorgangs aufgerufen. Die Delegates werden für jede Konfiguration aufgerufen, bei der eine Aktion durch den Anwender erforderlich ist. Weitere Informationen zur Handhabung von Rückrufen finden Sie unter Unterstützung von Callbacks (Seite 288). Bestimmte Konfigurationen enthalten nur eine Information, weshalb keine Aktion durch den Anwender erforderlich ist.

Die möglichen Typen von Ladekonfigurationen werden nachstehend aufgeführt.

Konfigurationsname	Beschreibung und Eigenschaften
DownloadConfiguration	• Basisklasse für alle Konfigurationen. • Enthält die einzige Eigenschaft DownloadConfiguration.Message : string (schreibgeschützte Eigenschaft, die die Konfigurationsmeldung enthält)
DownloadSelectionConfiguration	• Basisklasse für alle auswählbaren Konfigurationen. • Enthält keine weiteren Eigenschaften. In allen von dieser Klasse abgeleiteten untergeordneten Klassen muss eine Auswahl bereitgestellt werden.
DownloadCheckConfiguration	• Basisklasse für alle Konfigurationen, die geprüft oder nicht geprüft sein können. • Enthält die einzige Eigenschaft DownloadCheckConfiguration.Checked: bool<string> (Lese-/Schreib-Eigenschaft, die angibt, ob die Konfiguration geprüft oder ungeprüft ist)
DownloadPasswordConfiguration	• Basisklasse für alle Konfigurationen, die ein Passwort für den Ladevorgang benötigen. • Enthält eine einzige Methode zum Festlegen des Passworts. DownloadPasswordConfiguration.SetPassword (password: SecureString) : void

Der Datentyp der Konfigurationen wird nachstehend aufgeführt.

Konfiguration	Datentyp	Beschreibung und Aktion

DownloadSelection-Configuration	StartModules	`Set CurrentSelection:StopModulesSelections`. Verfügbare Enumerationswerte sind
		`NoAction (No action)`
		`StopAll (Stop all)`
		Diese Module werden zum Laden in ein Gerät gestoppt.

	StopModules	`Set CurrentSelection:StartModulesSelections`. Verfügbare Enumerationswerte:
		`NoAction (No action)`
		`StartModule (Start module)`
		Diese Module werden nach dem Ladevorgang gestartet.

	AllBlocksDownload	`Set CurrentSelection:AllBlocksDownloadSelections`. Verfügbarer Enumerationswert ist `DownloadAllBlocks (Download all blocks to the device)` Lädt Software in das Gerät

	OverwriteSystemData	`Set CurrentSelection:OverwriteSystemDataSelections`. Verfügbare Enumerationswerte sind
		`NoAction (No action)`
		`Overwrite (Download to device)`
		Löscht und ersetzt die vorhandenen Systemdaten im Zielspeicherort.

	ConsistentBlocksDownload	`Set CurrentSelection:ConsistentBlocksDownloadSelections`. Verfügbarer Enumerationswert ist `ConsistentDownload (Consistent download)` Lädt die Software in das Gerät.
	AlarmTextLibrariesDownload	`Set CurrentSelection:AlarmTextLibrariesDownloadSelections`. Verfügbare Enumerationswerte sind ConsistentDownload (Consistent download) NoAction (No action) Lädt alle Alarmtexte und Textlistentexte.

	ProtectionLevelChanged	Set CurrentSelection:ProtectionLevelChangedSelections. Verfügbare Enumerationswerte sind
		`NoChange (No change)`
		`ContinueDownloading (Continue downloading to the device)`
		Der CPU-Schutz wird auf die nächstniedrigere Stufe gesetzt.

	ActiveTestCanBeAborted	`Set CurrentSelection:ActiveTestCanBeAbortedSelections`. Verfügbare Enumerationswerte sind
		`NoAction (No action)`
		`AcceptAll (Accept all)`
		Aktive Test- und Inbetriebnahmefunktionen werden während des Ladebetriebs des Geräts abgebrochen.

	ResetModule	`Set CurrentSelection:ResetModuleSelections` Verfügbare Enumerationswerte sind

Konfiguration	Datentyp	Beschreibung und Aktion

		`NoAction (No action)`
		`DeleteAll (Delete all)`
		Setzt das Modul zurück.

	LoadIdentificationData	`Set CurrentSelection:LoadIdentificationDataSelections.` Verfügbare Enumerationswerte sind
		`LoadNothing (Load nothing)`
		`LoadData (Load data)`
		`LoadSelected (Load selected)`
		Identifikationsdaten in die PROFINET IO-Geräte und ihre Module laden.

	DifferentTargetConfiguration	`Set CurrentSelection:DifferentTargetConfigurationSelections.` Verfügbare Enumerationswerte sind
		`NoAction (No action)`
		`AcceptAll (Accept all)`
		Gibt den Unterschied zwischen konfiguriertem Modul und Zielmodul (online) an

	InitializeMemory	`Set CurrentSelection:InitializeMemorySelections.` Verfügbare Enumerationswerte:
		`NoAction (No action)`
		`AcceptAll (Accept all)`
		Dieser Datentyp dient zum Initialisieren des Speichers.

DownloadCheckConfiguration	CheckBeforeDownload	Eigenschaft `Set IsChecked:bool` Prüft vor dem Laden in das Gerät.
	UpgradeTargetDevice	Eigenschaft `Set IsChecked:bool` Prüft die unterschiedlichen Projektversionen im konfigurierten Gerät und im Zielgerät (online).
	OverWriteHMIData	Eigenschaft `Set IsChecked:bool` Überschreibt die Objekte online.
	FitHMIComponents	Eigenschaft `Set IsChecked:bool` Komponenten mit einer anderen Version werden auf dem Zielgerät installiert.
	TurnOffSequence	Eigenschaft `Set IsChecked:bool` Schaltet die Sequenz vor dem Laden aus.
	OverwriteTargetLanguages	Eigenschaft `Set IsChecked:bool` Zum Unterscheiden der Einstellungen für Projekt und PLC-Programmierung
	DowngradeTargetDevice	Eigenschaft `Set IsChecked:bool` Zum Erwähnen der verschiedenen Datenformate in Online- und Offline-Projekten.

Konfiguration	Datentyp	Beschreibung und Aktion
DownloadPassword-Configuration	ModuleReadAccessPassword	Methode `Set password via SetPassword(password:SecureString).` Passwort eingeben, um Lesezugriff auf das Modul zu erhalten.
	ModuleWriteAccessPassword	Methode `Set password via SetPassword(password:SecureString) .` Passwort eingeben, um Schreibzugriff auf das Modul zu erhalten.
	BlockBindingPassword	Methode `Set password via SetPassword(password:SecureString).` Methode für die Konfiguration eines bausteingebundenen Passworts.

ACHTUNG

Beachten Sie bitte, dass die Ladekonfigurationen den Konfigurationen in den Dialogen mit der Ladevorschau und den Ladeergebnissen ähneln, wenn Sie mit der Benutzeroberfläche des TIA Portals arbeiten.

⚠ WARNUNG

Der API-Benutzer ist verantwortlich für die Sicherheitsmaßnahmen zur Handhabung von Passwörtern durch Code.

Nicht verarbeitete Konfiguration, die den Ladevorgang verhindern kann, verursacht eine EngineeringTargetInvocationException und bricht den Ladevorgang ab. Bei einer nicht

verarbeiteten Ausnahme im Delegate wird eine EngineeringDelegateInvocationException ausgelöst.

Beispiel für die Implementierung von 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
   }
```

Beispiel für die Implementierung von PostDownloadDelegate:

```
 private static void PostConfigureDownload(DownloadConfiguration downloadConfiguration)
   {
               StartModules startModules = downloadConfiguration as StartModules;
               if (startModules != null)
                    {
                    startModules.CurrentSelection = StartModulesSelections.StartModule; //
Sets PLC in "Run" mode
                    }
   }
```

Parameter 4: DownloadOptions

Der Anwender muss die Ladeoptionen über DownloadOptions, durch enum gekennzeichnet, angeben. Dieser Parameter bestimmt den Typ des durchzuführenden Ladevorgangs, z. B. Hardware, Software oder Hardware und Software.

```
[Flags]
   public enum DownloadOptions   {
      None = 0,    // Download nothing
      Hardware,    // Download hardware only
     Software     // Download software only
   }
```

Wenn der Anwender sowohl Software als auch Hardware in das Gerät laden möchte, ist DownloadOptions.Hardware | DownloadOptions.Software als 4. Parameter der Methode Download zu übergeben.

DownloadResult

Das von der Aktion Download zurückgegebene DownloadResult liefert Rückmeldung über den Zustand der geladenen Objekte.

Beispiel für den Aufruf von 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);
      }
   }
```

Ändern Sie den folgenden Code, um durch Aufruf der entsprechenden Konfiguration in den PLC zu laden:

```
private static void DownloadNCU(Device ncu, ConfigurationTargetInterface
configurationTargetInterface)
 {
       DownloadConfigurationDelegate preDownloadDelegate = PreConfigureDownload;
       DownloadConfigurationDelegate postDownloadDelegate = PostConfigureDownload;
       DownloadProvider downloadProvider = null;
        foreach (var item in ncu.DeviceItems[0].DeviceItems)
            {
         downloadProvider = item.GetService<DownloadProvider>();
                if (downloadProvider != null)
                {
                    break;
                }
            }
          downloadProvider.Configuration.ApplyConfiguration(configurationTargetInterface);
            IConfiguration targetConfiguration = configurationTargetInterface;
           downloadProvider.Download(targetConfiguration, preDownloadDelegate,
postDownloadDelegate, DownloadOptions.Hardware | DownloadOptions.Software);
}
```

7.18.6.2 PLC starten und stoppen

Voraussetzungen

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Der PLC ist offline.

Verwendung

Bei Interaktionen mit TIA Portal über Openness API kann es notwendig sein, die Betriebsart des PLC zu ändern. TIA Openness bietet die Möglichkeit, den Betriebszustand des PLC entweder in Start oder in Stop zu ändern.

Programmcode

Ändern Sie folgenden Programmcode, um den Betriebszustand des PLC auf STOP einzustellen.

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
 {
    StopModules stopModules = downloadConfiguration as StopModules;
    if (stopModules != null)
         {
         // Puts PLC in "Stop" mode
         stopModules.CurrentSelection = StopModulesSelections.StopAll;
    }
}
```

Ändern Sie folgenden Programmcode, um den Betriebszustand des PLC auf START einzustellen.

```
public void ConfigurePostDownload(DownloadConfiguration downloadConfiguration)
  {
   StartModules startModules = downloadConfiguration as StartModules;
      if (startModules != null)
         {
          // Puts PLC in "Start" mode
          startModules.CurrentSelection = StartModulesSelections.StartModule;
    }
 }
```

7.18.6.3 Unterstützung von Callbacks

Voraussetzungen

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Bestimmte API-Methoden erfordern während ihrer Ausführung eine Interaktion mit dem benutzerdefinierten Anwendungscode. Delegates dienen zur Handhabung dieser Callback-Aktionen im benutzerdefinierten Anwendungscode. Sie müssen eine Methode mit einer kompatiblen Signatur implementieren und als Parameter delegate an die Aktion übergeben. Für die Ausführung ruft das TIA Portal die implementierten Methoden auf.

Programmcode

```
// This delegate is declared in Siemens.Engineering.dll
public delegate void
Siemens.Engineering.Download.DownloadConfigurationDelegate(Siemens.Engineering.Download.Co
nfigurations.DownloadConfiguration configuration);
...
```

Beispiel für einen benutzerdefinierten Anwendungscode mit Verwendung und Implementierung von delegate:

```
[STAThread]
static void Main()
{
  ...
    DownloadProvider downloadProvider = ...;
    IConfiguration targetConfiguration = ...;
    DownloadConfigurationDelegate preDownloadDelegate = PreConfigureDownload;
    DownloadConfigurationDelegate postDownloadDelegate = PostConfigureDownload;
    DownloadResult result = downloadProvider.Download(targetConfiguration,
preDownloadDelegate, postDownloadDelegate, DownloadOptions.Hardware |
DownloadOptions.Software);
  ...

}

//This method will be called back by TIA Portal
private static void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
   {
     // Work with the parameter
   }

//This method will be called back by TIA Portal
private static void ConfigurePostDownload(DownloadConfiguration downloadConfiguration)
    {
      // Work with the parameter
    }
```

Hinweis

Das Attribut **STAThread** stellt sicher, dass die Delegates im Haupt-Thread der Ausführung aufgerufen werden.

7.18.6.4 PLC durch Passwort schützen

Voraussetzungen

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)
- Der PLC ist offline.

Verwendung

Bei Interaktionen mit TIA Portal über Openness API kann es notwendig sein, den Schutzgrad des PLC zu ändern. TIA Openness bietet die Möglichkeit, den PLC durch ein Passwort zu schützen. Das Passwort kann sowohl für lesegeschützte als auch für schreibgeschützte PLCs eingestellt werden.

Programmcode

Ändern Sie folgenden Programmcode für lesegeschützte PLCs.

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
  {
       ModuleReadAccessPassword moduleReadAccessPassword = downloadConfiguration
asModuleReadAccessPassword;
       if (moduleReadAccessPassword != null)
          {
            SecureString password = ...; // Get password from a secure location
            moduleReadAccessPassword.SetPassword(password); // enter the password to gain
full access
          }
  }
```

Ändern Sie den folgenden Programmcode für schreibgeschützte PLCs:

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
  {
    ModuleWriteAccessPassword moduleWriteAccessPassword = downloadConfigurationas
ModuleWriteAccessPassword;
    if (moduleWriteAccessPassword != null)
     {
       SecureString password = ...; // Get password from a secure location
      moduleWriteAccessPassword.SetPassword(password); // enter the password to gain full
access
     }
  }
```

⚠ WARNUNG
Der API-Benutzer ist verantwortlich für die Sicherheitsmaßnahmen zur Handhabung von Passwörtern durch Code.

7.18.6.5 Handhabung bausteingebundener PLC-Passwörter

Voraussetzungen

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Der PLC ist offline.

Verwendung

TIA Openness unterstützt die Datenbindung von Passwörtern für Kundenanwendungen. TIA Openness bietet dem Kunden einen Weg, ein bausteingebundenes Passwort anzugeben. Zum Beispiel kann ein bausteingebundenes Passwort auf der Klasse DownloadPasswordConfiguration durch Aufrufen der Methode SetPassword konfiguriert werden.

Hinweis

Wenn Sie den Ladevorgang mit einem password schützen möchten, muss bei jeden Aufruf der Download-Funktion ein password angegeben werden. Das gilt unabhängig davon, ob das Gerät bereits konfiguriert ist. Nach erfolgreicher Eingabe des Passworts für eine gegebene Konfiguration werden alle nachfolgenden Aufrufe von SetPassword ignoriert.

Programmcode

Ändern Sie folgenden Programmcode:

```
public void ConfigurePreDownload(DownloadConfiguration downloadConfiguration)
{
    DownloadPasswordConfiguration downloadPasswordConfiguration = downloadConfiguration as
DownloadPasswordConfiguration;
    if(downloadPasswordConfiguration != null &&
downloadPasswordConfiguration.Message.Contains("block_1"))
       {
              SecureString password = ...; // Get password from a secured location
              downloadPasswordConfiguration.SetPassword(password);
    }
}
```

7.18.7 Hardware, Software und Dateien in PLC-Gerät laden

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Der Openness-Anwender kann über StationUploadProvider (auf das von einem project aus zugegriffen wird) eine Station in ein Projekt laden. Das Laden in eine DeviceGroup wird nicht unterstützt. Wenn ein project zur Durchführung eines Ladevorgangs verwendet wird, gibt der Dienst bei Aufruf von GetService eine Instanz von StationUploadProviderzurück. Andernfalls wird null zurückgegeben.

Programmcode: StationUploadProvider-Dienst von einem Projekt abrufen

```
Project myProject = ...;
StationUploadProvider uploadProviderForProject =
myProject.GetService<StationUploadProvider>();
if (uploadProviderForProject != null)
{
    ...
}
```

Parameter der Upload-Methode

Um das Laden in ein PLC-Gerät durchzuführen, ruft der Anwender die Upload-Methode StationUpload von StationUploadProvider. auf. Die Parameter dieser Upload-Methode sind ConfigurationAddress und UploadConfigurationDelegte . Da mit der Methode StationUpload Software, Hardware und Dateien geladen werden, sind die UploadOptions optional.

Parametername	Typ	Beschreibung
configurationAddress	Siemens.Engineering.Connection.ConfigurationAddress	Adresse des zu ladenden Geräts
uploadConfigurationDelegate	Siemens.Engineering.Upload.UploadConfigurationDelegate	Delegate, der aufgerufen wird, um die Konfiguration vor dem Laden zu prüfen
uploadOptions	Siemens.Engineering.Upload.UploadOptions	Upload-Optionen

Parameter 1: ConfigurationAddress

Der Anwender muss das Objekt ConfigurationAddress für die Upload-Methode angeben. Das Adressobjekt dient dazu, eine Verbindung zum angegebenen PLC-Gerät herzustellen, in das geladen werden soll. Das Objekt ConfigurationAddress muss in der ConnectionConfiguration des StationUploadProvider. erstellt werden. Die Configuration enthält eine Liste unterstützter Modes. Einer der Modes muss zur Verwendung beim Laden ausgewählt werden. Der ausgewählte ConfigurationMode enthält eine Liste aller lokalen PcInterfaces, die den ausgewählten Mode unterstützen. Eine dieser Schnittstellen muss ausgewählt werden. Die gewünschte Adresse kann in der Address -Sammlung der ausgewählten ConfigurationPcInterface erstellt werden.

Ändern Sie zum Erstellen eines Adressobjekts den folgenden Code:

```
...
StationUploadProvider uploadProvider = null;
...
ConnectionConfiguration configuration = uploadProvider.Configuration;
ConfigurationMode configurationMode = configuration.Modes.Find("PN/IE");
ConfigurationPcInterface pcInterface = configurationMode.PcInterfaces.Find("Intel(R)
Ethernet Connection I217-LM", 1);
//"Create an address. This "ConfigurationAddress" is used as parameter for upload."
ConfigurationAddress uploadAddress = pcInterface.Addresses.Create("192.68.0.1");
...
```

Upload des Projekts

Der Anwender kann den Stationsupload durch Aufrufen der Aktion StationUpload starten.

Die folgenden Parameter sind obligatorisch:

- ConfigurationAddress: Die Adresse des Geräts, in das geladen werden soll
- UploadConfigurationDelegate: Der Rückruf zur Behandlung von Lade-Inhibits

```
...
StationUploadProvider uploadProvider = null;
Device uploadedObject = null;
...
UploadConfigurationDelegate preUploadDelegate = PreConfigureUpload;
UploadResult result = uploadProvider.StationUpload(uploadAddress, preUploadDelegate);
 // The uploaded device
uploadedObject = result.UploadedStation;
if (uploadedObject == null)
{
 ...
}
internal void PreConfigureUpload(UploadConfiguration uploadConfiguration)
{
 ...
}
```

Parameter2: UploadConfigurationDelegate

Der Openness-Anwender muss eine Implementierung von void UploadConfigurationDelegate (UploadConfiguration uploadConfiguration) bereitstellen. Der Delegate wird für Konfigurationen vor dem Laden aufgerufen. Der Delegate wird für jede Konfiguration aufgerufen, bei der eine Aktion durch den Anwender erforderlich ist. Weitere Informationen zur Handhabung von Rückrufen finden Sie unter Unterstützung von Callbacks (Seite 288). Bestimmte Konfigurationen enthalten nur eine Information, weshalb keine Aktion durch den Anwender erforderlich ist.

Die möglichen Typen von Ladekonfigurationen werden nachstehend aufgeführt:

Konfigurationsname	Beschreibung und Eigenschaften
UploadConfiguration	• Basisklasse für alle Konfigurationen. Sie enthält Informationen im Message-Attribut • Enthält die einzige Eigenschaft UploadConfiguration.Message : string (schreibgeschützte Eigenschaft, die die Konfigurationsmeldung enthält)
UploadPasswordConfiguration	• Abgeleitet von UploadConfiguration • Basisklasse für alle Konfigurationen, die ein Passwort für den Ladevorgang benötigen. • Enthält eine einzige Methode zum Festlegen des Passworts. UploadPasswordConfiguration.SetPassword (password: SecureString) : void - Passwort festlegen
UploadSelectionConfiguration	• Abgeleitete Klasse von UploadConfiguration • Enthält keine weiteren Eigenschaften.

Der Datentyp der Konfigurationen wird nachstehend aufgeführt.

Konfiguration	Datentyp	Beschreibung und Aktion
UploadPasswordConfiguration	ModuleReadAccessPassword	Methode `Set password via SetPassword(password:SecureString).` Passwort eingeben, um Lesezugriff auf das Modul zu erhalten.
	PasswordReadAccess	Methode `Set password via SetPassword(password:SecureString).` Passwort für den Softwareupload in klassischen PLCs eingeben, um Lesezugriff auf das Modul zu erhalten.

UploadSelectionConfiguration	UploadMissingProducts	`Set CurrentSelection:UploadMissingProductsSelections`
		Verfügbare Enumerationswerte:
		`TryUpload (Consistent upload)`
		`NoAction (No action)`
		Eine Auswahl für den Upload angeben.

Unterstützung eines fehlersicheren Passworts ist nicht notwendig. Für den Lesezugriff beim Laden in einen F-PLC wird kein Passwort benötigt.

Eine nicht behandelte Konfiguration, die den Ladevorgang verhindern kann, verursacht eine EngineeringTargetInvocationException und bricht den Ladevorgang ab.

Bei einer nicht behandelten Ausnahme im Delegate wird eine EngineeringDelegateInvocationException ausgelöst.

Beispiel für die Implementierung von PreUploadDelegate:

```
private static void PreConfigureUpload(UploadConfiguration UploadConfiguration)
 {
 ModuleReadAccessPassword moduleReadAccessPassword = UploadConfiguration as
ModuleReadAccessPassword;
 if (moduleReadAccessPassword != null)
 {
 string passWD = "passWD";
 var password = new SecureString();
 foreach (var c in passWD)
 password.AppendChar(c);

 moduleReadAccessPassword.SetPassword(password);
 return;
 }


 ModuleWriteAccessPassword moduleWriteAccessPassword = UploadConfiguration as
ModuleWriteAccessPassword;
 if (moduleWriteAccessPassword != null)
 {
 string passWD = "passWD";
 var password = new SecureString();
 foreach (var c in passWD)
 password.AppendChar(c);

 moduleWriteAccessPassword.SetPassword(password);
 return;
 }
 ...

 throw new NotSupportedException(); // Exception thrown in the delagate will cancel upload
 }
```

Parameter3: UploadOptions

Die Upload-Optionen kann der Anwender nicht angeben. Die Upload-Optionen lauten: "Hardware", "Software", "Hardware und Software" sowie "Hardware, Software und Dateien".

UploadResult

Das von der Upload-Aktion zurückgegebene Ergebnis UploadResult liefert Rückmeldung über den Zustand der geladenen Objekte.

- UploadResult.Message: UploadResultMessageComposition - Zusammensetzung von UploadResultMessage

Die folgenden Attribute werden unterstützt:

Attribute	Beschreibung
ErrorCount	int-Wert der Fehler beim Upload
State	UploadResultState mit den möglichen Werten: Success, Information, Warning und Error
UploadedStation	Eine Device-Instanz der geladenen Station
WarningCount	Anzahl der Warnungen beim Laden als int

Die UploadResultMessage enthält:

- UploadResultMessage.Messages : UploadResultMessageComposition - Zusammensetzung von UploadResultMessage

Die folgenden Attribute werden unterstützt:

Attribute	Beschreibung
DateTime	System.DateTime der erstellten Meldung.
ErrorCount	Ein int-Zähler für Fehler.
State	UploadResultState mit den möglichen Werten: Success, Information, Warning und Error
WarningCount	Anzahl der Warnungen beim Laden als int

Beispiel für den Aufruf von Upload

```
internal bool UploadPLC()
 {
 ...
 UploadResult result = uploadProvider.StationUpload(uploadAddress, preUploadDelegate);
 ...
 PrintAllMessages(result.Messages, 0);
 ...
 }


 internal void PrintAllMessages(UploadResultMessageComposition messages, int level)
 {
 if (messages == null)
 return;

 if (level == 0)
 Console.WriteLine("\n");
 foreach (UploadResultMessage message in messages)
 {
 string messageOut = message.Message.PadLeft(message.Message.Length + level, '\t') + "\n";
 Console.WriteLine(messageOut);

 if ((message.Messages != null) && (message.Messages.Count > 0))
 PrintAllMessages(message.Messages, level+1);
 }
 }
```

7.18.8 Zugriff auf Fingerabdrücke

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Anwendung

Über die TIA Portal Openness API können Sie Änderungen in Bausteinen oder UDTs erkennen. Hierfür können Sie die Fingerabdrücke des Objekts miteinander vergleichen. Eine Fingerabdruck-Instanz enthält eine FingerprintId, die die Art des Fingerabdrucks festlegt, sowie den fingerprint-Wert als String. Alle bereitgestellten fingerprints berücksichtigen nur Benutzereingaben, kein Übersetzungsergebnis oder andere Änderungen, die das System vornimmt.

Die Enumeration FingerprintId listet alle Arten von Fingerabdrücken auf, die in Openness unterstützt werden:

Wert	Beschreibung
Code	Berücksichtigt alle Änderungen am Code im Hauptteil des Bausteins. Das Übersetzungsergebnis wird nicht berücksichtigt.
Interface	Berücksichtigt alle Änderungen an der Schnittstelle eines Bausteins, einschließlich der Startwerte eines DB.
Properties	Berücksichtigt Änderungen an den Eigenschaften eines Bausteins, z. B. Name, Nummer.
Comments	Berücksichtigt Änderungen an den Kommentaren eines Bausteins. Bei OBs ändert sich der Fingerabdruck auch, wenn die Liste verfügbarer Sprachen in den Spracheinstellungen des Projekts geändert wird.
LibraryType	Vorhanden, wenn ein Baustein mit einem Bibliothekstyp verschaltet ist.
Texts	Bei V15 SP1 ist dieser Fingerabdruck nur für Graph-Bausteine vorhanden.
Alarms	Vorhanden, wenn ein Baustein mit der Meldungsfunktion arbeitet.
Supervision	Vorhanden, wenn ein Baustein eine Überwachungsfunktion enthält.
TechnologyObject	Nur für Technologieobjekt-DBs vorhanden.
Events	Nur für OBs vorhanden.
TextualInterface	Vorhanden, wenn der Baustein eine Textschnittstelle besitzt.

Programmcode

Sie benötigen den FingerprintProvider , um die Fingerabdrücke eines Objekts aufzurufen. Dieser Dienst ist für Bausteine (FB, FC, OB, DB) und UDTs, jedoch nicht für Variablentabellen verfügbar. Der FingerprintProvider berechnet alle verfügbaren Fingerabdrücke eines Objekts und gibt diese bei jedem Aufruf von GetFingerprints zurück. Um korrekte Fingerabdrücke zu erhalten, muss der Baustein oder UDT vor dem Aufrufen der Fingerabdrücke konsistent sein. Sonst wird eine RecoverableException gemeldet. Ist ein Fingerabdruck nach seiner Berechnung ungültig, wird eine RecoverableException gemeldet.

```
PlcBlock block = ...;
FingerprintProvider provider = block.GetService<FingerprintProvider>();
IList<Fingerprint> fingerprints = provider.GetFingerprints();
foreach(var fingerprint in fingerprints)
{
 string fpValue = fingerprint.Value;
 FingerprintId fpId = fingerprint.Id;
}
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

7.18.9 PLC-Software vergleichen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Sie haben die folgenden Möglichkeiten, um die Abweichung zwischen der Software zweier Geräte zu ermitteln:

- Vergleich der Software zweier konfigurierter PLCs
- Vergleich der Software eines PLC und der Projektbibliothek
- Vergleich der Software eines PLC und der globalen Bibliothek
- Vergleich der Software eines PLC und der Masterkopie eines PLC
- Vergleich der Software eines konfigurierten PLC mit der Software eines verbundenen PLC im Status „online"

Signatur

Verwenden Sie die Methode `CompareTo` oder `CompareToOnline` für den Vergleich.

```
public CompareResult CompareTo (ISoftwareCompareTarget compareTarget)
public CompareResult CompareToOnline ()
```

Rückgabewert/Parameter	Funktion
`CompareResult compareResult`	Gibt das Vergleichsergebnis zurück: • `FolderContentsDifferent`: Inhalt der verglichenen Ordner unterscheidet sich. • `FolderContentsIdentical`: Inhalt der verglichenen Ordner ist identisch. • `ObjectsDifferent`: Inhalt der verglichenen Objekte unterscheidet sich. • `ObjectsIdentical`: Inhalt der verglichenen Objekte ist identisch. • `LeftMissing`: Das Objekt ist nicht in dem Objekt enthalten, von dem aus der Vergleich gestartet wurde. • `RightMissing`: Das Objekt ist nicht in dem Objekt enthalten, das verglichen wird.
`ISoftwareCompareTarget compareTarget`	Liste vergleichbarer Objekte.

Programmcode

Ändern Sie den folgenden Programmcode, um das Vergleichsergebnis auszugeben:

```
private static void WriteResult(CompareResultElement compareResultElement, string indent)
{
    Console.WriteLine("{0} <{1}> <{2}> <{3}> <{4}> ",
    indent,
    compareResultElement.LeftName,
    compareResultElement.ComparisonResult,
    compareResultElement.RightName,
    compareResultElement.DetailedInformation);
    WriteResult(compareResultElement.Elements, indent);
}
private static void WriteResult (IEnumerable<CompareResultElement> compareResultElements,
string indent)
{
    indent += " ";
    foreach (CompareResultElement compareResultElement in compareResultElements)
    {
        WriteResult(compareResultElement, indent);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software von Geräten zu vergleichen:

```
private static void CompareTwoOfflinePlcs(PlcSoftware plcSoftware0, PlcSoftware
plcSoftware1)
{
    if (plcSoftware0 != null && plcSoftware1 != null)
    {
        CompareResult compareResult = plcSoftware0.CompareTo(plcSoftware1);
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software eines PLC mit der Projektbibliothek zu vergleichen:

```
private static void ComparePlcToProjectLibrary(Project project, PlcSoftware plcSoftware)
{
    if (project != null && plcSoftware != null)
    {
        CompareResult compareResult = plcSoftware.CompareTo(project.ProjectLibrary);
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software eines PLC mit der globalen Bibliothek zu vergleichen:

```
private static void ComparePlcToGlobalLibrary(PlcSoftware plcSoftware, GlobalLibrary
globalLibrary)
{
    if (plcSoftware != null && globalLibrary != null)
    {
        CompareResult compareResult = plcSoftware.CompareTo(globalLibrary);
        WriteResult(compareResult.RootElement, String.Empty);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software eines PLC mit einer Masterkopie zu vergleichen:

```
private static void ComparePlcToMasterCopy(Project project, PlcSoftware plcSoftware)
{
    if (project != null && plcSoftware != null)
    {
        CompareResult compareResult =
plcSoftware.CompareTo(project.ProjectLibrary.MasterCopyFolder.MasterCopies[0]);
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

Ändern Sie den folgenden Programmcode, um die Software eines PLC mit der Software eines verbundenen PLC zu vergleichen:

```
private static void ComparePlcToOnlinePlc(PlcSoftware plcSoftware)
{
    if (plcSoftware != null)
    {
        CompareResult compareResult = plcSoftware.CompareToOnline();
        WriteResult(compareResult.RootElement, string.Empty);
    }
}
```

7.18.10 PLC-Hardware vergleichen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Sie haben mit Ihrer TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Anwendung

Mithilfe der TIA Openness API können Sie die Hardware zweier PLC-Geräte vergleichen.

Signatur

Verwenden Sie die Methode CompareTo für den Vergleich der beiden Hardwareobjekte.

CompareResult CompareTo (IHardwareCompareTarget compareTarget);

Rückgabewert/-parameter	Funktion
CompareResult compare result	Das Vergleichsergebnis zurückgeben: • FolderContainsDifferencesOwnStateDifferent: Der Inhalt des Ordners weist einen oder zwei Unterschiede auf, der eigene Zustand des Ordners weicht ab. • FolderContentEqualOwnStateDifferent: Der Ordnerinhalt ist identisch, der eigene Zustand des Ordners weicht ab.
IHardwareCompareTarget compareTarget	Das Vergleichsziel, wegen dessen der Hardwarevergleich durchgeführt wird. Darf nicht null sein.

Wenn der Parameter compareTarget null ist und ein Hardwarevergleich versucht wird, wird die Siemens.Enginnering.EngineeringTargetInvocationExceptions ausgelöst.

Programm

Ändern Sie den folgenden Programmcode, um das Vergleichsergebnis auszugeben:

```
...
CompareResult compareResult = plc_1.CompareTo(plc_2);
CompareResultState resultState = compareResult.RootElement.ComparisonResult;
if (resultState == CompareResultState.FolderContainsDifferencesOwnStateDifferent)
{
 // Folder contents have one or more differences, folder's own state is different:
 // May occur if the plc has a different subordinate element, e.g., a local module, and
 // the plc itself is different, e.g., in a parameter
}
else if (resultState == CompareResultState.FolderContentEqualOwnStateDifferent)
{
 // Folder content is the same, folder's own state is different:
 // May occur if a folder-style module, e.g., FM 351, has equal subordinate elements but
 // the module itself is different, e.g., in a parameter
}
else if (resultState == CompareResultState.FolderContentsIdentical)
{
 ...
}
...
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

7.18.11 Online-Verbindung zur PLC aufbauen oder trennen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Alle Geräte werden enumeriert.
  Siehe Geräteelemente aufrufen (Seite 236).

Verwendung

Sie können die Online-Verbindung zu einer PLC herstellen oder eine bestehende Online-Verbindung trennen.

Programmcode

Um die Online-Verbindung zu einer PLC herzustellen oder zu trennen, ändern Sie folgenden Programmcode:

```
public static void SetOnlineConnection(DeviceItem deviceItem)
{
    OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
    if (onlineProvider == null) { return; }
    // Go online
    if (onlineProvider.Configuration.IsConfigured)
    {
        onlineProvider.GoOnline();
    }
   // Go offline
   onlineProvider.GoOffline();
}
```

Sie können auch die Online-Verbindungen zu allen verfügbaren PLCs in einem Projekt herstellen oder trennen.

```
public static void SetOnlineConnectionForAllPLCs(Project project)
{
    foreach (Device device in project.Devices)
    {
        foreach (DeviceItem deviceItem in device.DeviceItems)
        {
            OnlineProvider onlineProvider = deviceItem.GetService<OnlineProvider>();
            if (onlineProvider != null)
            {
                // Establish online connection to PLC:
                onlineProvider.GoOnline();

                // ...

                // Disconnect online connection to PLC:
                onlineProvider.GoOffline();
            }
        }
    }
}
```

7.18.12 Bausteine

7.18.12.1 Gruppe "Programmbausteine" abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.

Programmcode

Ändern Sie den folgenden Programmcode, um die Gruppe "Programmbausteine" abzufragen:

```
private static void GetBlockGroupOfPLC(PlcSoftware plcsoftware)
//Retrieves the system group of a block
{
    PlcBlockSystemGroup blockGroup = plcsoftware.BlockGroup;
}
```

7.18.12.2 Systemgruppe für Systembausteine abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode:

Ändern Sie den folgenden Programmcode, um die für Systembausteine vom System erzeugte Gruppe zu ermitteln:

```
PlcSoftware plcSoftware = ...
foreach (PlcSystemBlockGroup systemGroup in plcSoftware.BlockGroup.SystemBlockGroups)
{
    foreach (PlcSystemBlockGroup group in systemGroup.Groups)
    {
        PlcBlockComposition pbComposition = group.Blocks;
        foreach (PlcBlock block in pbComposition)
        {
            //Fügen Sie Ihren Code hier hinzu
        }
    }
}
```

7.18.12.3 Systemuntergruppen enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode: Alle Systemuntergruppen enumerieren

Ändern Sie den folgenden Programmcode, um die Systemuntergruppen aller Systembausteine zu enumerieren:

```
//Retrieves the system generated group for system blocks
private static void GetSystemgroupForSystemblocks(PlcSoftware plcSoftware)
{
    PlcSystemBlockGroupComposition systemBlockGroups =
plcSoftware.BlockGroup.SystemBlockGroups;
    if (systemBlockGroups.Count != 0)
    {
        PlcSystemBlockGroup sbSystemGroup = systemBlockGroups[0];
        foreach (PlcSystemBlockGroup group in sbSystemGroup.Groups)
        {
            EnumerateSystemBlockGroups(group);
        }
    }
}
private static void EnumerateSystemBlockGroups(PlcSystemBlockGroup systemBlockGroup)
{
    foreach (PlcSystemBlockGroup group in systemBlockGroup.Groups)
    {
        // recursion EnumerateSystemBlockGroups(group);
    }
}
```

Programmcode: Auf eine bestimmte Untergruppe zugreifen

Ändern Sie den folgenden Programmcode, um auf eine bestimmte Untergruppe zuzugreifen:

```
private static void AccessSbGroup(PlcSystemBlockGroup systemBlockGroup)
{
    PlcSystemBlockGroup group1 = systemBlockGroup.Groups.Find("User group XYZ");
    PlcSystemBlockGroup group2 = group1.Groups.Find("User group ZYX");
}
```

Siehe auch

Externe Datei hinzufügen (Seite 319)

7.18.12.4 Benutzerdefinierte Bausteingruppen enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.

Verwendung

Enthaltene Untergruppen werden beim Enumerieren rekursiv berücksichtigt.

Programmcode: Alle Gruppen enumerieren

Ändern Sie den folgenden Programmcode, um die benutzerdefinierten Bausteingruppen zu enumerieren:

```
//Enumerates all block user groups including sub groups
private static void EnumerateAllBlockGroupsAndSubgroups(PlcSoftware plcsoftware)
{
    foreach (PlcBlockUserGroup blockUserGroup in plcsoftware.BlockGroup.Groups)
    {
        EnumerateBlockUserGroups(blockUserGroup);
    }
}

private static void EnumerateBlockUserGroups(PlcBlockUserGroup blockUserGroup)
{
    foreach (PlcBlockUserGroup subBlockUserGroup in blockUserGroup.Groups)
    {
        EnumerateBlockUserGroups(subBlockUserGroup);
        // recursion
    }
}
```

Programmcode: Auf eine Gruppe zugreifen

Ändern Sie den folgenden Programmcode, um auf eine ausgewählte benutzerdefinierte Bausteingruppe zuzugreifen:

```
//Gives individual access to a specific block user group
private static void AccessBlockusergroup(PlcSoftware plcsoftware)
{
    PlcBlockUserGroupComposition userGroupComposition = plcsoftware.BlockGroup.Groups;
    PlcBlockUserGroup plcBlockUserGroup = userGroupComposition.Find("MyUserfolder");
}
```

7.18.12.5 Alle Bausteine enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.

Verwendung

Der gezielte Zugriff auf einen Programmbaustein ist möglich, wenn dessen Name bekannt ist.

Programmcode: Alle Bausteine enumerieren

Ändern Sie den folgenden Programmcode, um die Bausteine aller Bausteingruppen zu enumerieren:

```
private static void EnumerateAllBlocks(PlcSoftware plcsoftware)
//Enumerates all blocks
{
    foreach (PlcBlock block in plcsoftware.BlockGroup.Blocks)
    {
        // Do something...
    }
}
```

Programmcode: Auf einen bestimmten Baustein zugreifen

Um auf einen bestimmten Baustein zuzugreifen, ändern Sie folgenden Programmcode:

```
private static void AccessASingleBlock(PlcSoftware plcsoftware)
//Gives individual access to a block
{
    // The parameter specifies the name of the block
    PlcBlock block = plcsoftware.BlockGroup.Blocks.Find("MyBlock");
}
```

7.18.12.6 Informationen eines Bausteins/Anwenderdatentyps abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die TIA Portal Openness API unterstützt die Abfrage der folgenden Informationen für Programm und Datenbausteine und für Anwenderdatentypen:

- Zeitstempel im UTC-Zeitformat
  Über den Zeitstempel erfahren Sie Folgendes:
  - Wann der Baustein zuletzt übersetzt wurde
  - Wann der Baustein zuletzt geändert wurde
- Attribut „Consistency“
  Das Attribut „Consistency" ist in folgenden Fällen auf „True" gesetzt:
  - Der Baustein wurde erfolgreich übersetzt.
  - Der Baustein wurde seit der Übersetzung nicht geändert.
  - An externen Objekten wurden keine Änderungen vorgenommen, die eine Neuübersetzung erforderlich machen würden.
- Verwendete Programmiersprache (nur Programm und Datenbausteine)
- Bausteinnummer
- Bausteinname
- Bausteinautor
- Bausteinfamilie
- Bausteintitel
- Bausteinversion

Weitere Informationen hierzu siehe Bausteine und Typen des TIA Portal Openness-Objektmodells (Seite 56).

Programmcode

Um die oben aufgeführten Informationen abzufragen, ändern Sie folgenden Programmcode:

```
private static void GetPlcBlockInformation(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    // Read information
    DateTime compileDate = plcBlock.CompileDate;
    DateTime modifiedDate = plcBlock.ModifiedDate;
    bool isConsistent = plcBlock.IsConsistent;
    int blockNumber = plcBlock.Number;
    string blockName = plcBlock.Name;
    ProgrammingLanguage programmingLanguage = plcBlock.ProgrammingLanguage;
    string blockAuthor = plcBlock.HeaderAuthor;
    string blockFamily = plcBlock.HeaderFamily;
    string blockTitle = plcBlock.HeaderName;
    System.Version blockVersion = plcBlock.HeaderVersion;
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.18.12.7 Einen Baustein schützen und Schutz aufheben

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Verwendung

Über die Klasse ProtectionProvider und den Dienst ProtectionProvider können Sie einen Baustein mit einem Passwort schützen und den Passwortschutz wieder aufheben. Der Dienst ProtectionProvider ist bei Bausteinen verfügbar, die folgende Voraussetzungen erfüllen:

- Der Baustein kann einen Know-how-Schutz erhalten
- Der Baustein ist ein Code-Baustein oder eine globale DB
- Der Baustein ist im aktuellen PLC unterstützt oder editierbar
- Der Baustein befindet sich nicht in einem schreibgeschützten Kontext
- Der Baustein ist nicht Know-how-geschützt
- Der Baustein ist nicht online
- Der Baustein ist keine CPU-DB

- Der Baustein nutzt keine klassische Verschlüsselungssprache, ProDiag oder ProDiag-OB
- Der Baustein ist kein verschlüsselter importierter klassischer Baustein

Falls der Baustein nicht alle Bedingungen erfüllt, wird von der Methode GetService() eine Nullreferenz ausgegeben.

Programmcode: Vorgänge in Bezug auf Know-how-Schutz ausführen

Ändern Sie folgenden Programmcode:

```
PlcBlock block = ...;

ProtectionProvider protectionProvider = block.GetService<ProtectionProvider>();
if (protectionProvider != null)
{
    ... // perform know-how protection related operations here
}
```

Einen Baustein schützen

Legen Sie mit der Methode Protect() ein Passwort fest, um den Programmierbaustein zu schützen.

```
void Protect(SecureString password)
```

Ein Fehler tritt auf,

- wenn Sie einen bereits geschützten Baustein zu schützen versuchen: Eine EngineeringTargetInvocationException wird mit einer Meldung ausgelöst, die besagt, dass Sie kein bereits geschütztes Objekt schützen können.
- wenn Sie einen Baustein mit einem leeren String als Passwort zu schützen versuchen: Eine EngineeringTargetInvocationException wird mit einer Meldung ausgelöst, die besagt, dass kein Passwort angegeben wurde.
- wenn Sie einen fehlersicheren Baustein zu schützen versuchen, während das Failsafe-Programm passwortgeschützt ist: Eine EngineeringTargetInvocationException wird ausgelöst.
- wenn Sie einen fehlersicheren Baustein zu schützen versuchen, während dieser Baustein nicht aufgerufen ist: Eine EngineeringTargetInvocationException wird ausgelöst.

Ungeschützter Baustein

Entfernen Sie das Passwort, mit dem der Programmierbaustein geschützt ist, mithilfe der Methode Unprotect().

```
void Unprotect(SecureString password)
```

Ein Fehler tritt auf,

- wenn Sie den Schutz eines bereits ungeschützten Bausteins aufzuheben versuchen: Eine EngineeringTargetInvocationException wird mit einer Meldung ausgelöst, die besagt, dass Sie nicht den Schutz eines ungeschützten Objekts aufheben können.
- wenn Sie den Schutz eines Bausteins mit dem falschen Passwort aufzuheben versuchen: Eine EngineeringTargetInvocationException wird mit einer Meldung ausgelöst, die besagt, dass das Passwort nicht akzeptiert wurde.
- wenn Sie einen Baustein mit einem leeren String als Passwort zu schützen versuchen: Eine EngineeringTargetInvocationException wird mit einer Meldung ausgelöst, die besagt, dass kein Passwort angegeben wurde.

Nach ungültigen Zeichen suchen

Da Sie einen Baustein unter Verwendung der Protect()-Methode mit beliebigen Zeichen schützen können, einschließlich Rückschritt, Tab usw., ist es unter Umständen nicht möglich, den Schutz im TIA Portal aufzuheben. Da Passwörter als SecureString übermittelt werden, müssen Sie selbst überprüfen, ob das Passwort ungültige Zeichen enthält. Mit der GetInvalidPasswordCharacters()-Methode können Sie eine Liste ungültiger Zeichen abrufen.

```
SecureString CreatePasswordString(ProtectionProvider protectionProvider, IEnumerable<char>
contentCharacters)
{
    IList<char> invalidCharacters = protectionProvider.GetInvalidPasswordCharacters();
    SecureString password = new SecureString();
    foreach(char ch in contentCharacters)
    {
        if (!invalidCharacters.Contains(ch))
        {
            password.AppendChar(ch);
        }
        else
        {
            // at least one of the content characters is not valid
            // signal an error - e.g. throw an exception
            ...
        }
    }
    return password;
}
```

Ein Fehler tritt auf,

- wenn Sie den Schutz eines bereits ungeschützten Bausteins aufzuheben versuchen: Eine EngineeringTargetInvocationException wird mit einer Meldung ausgelöst, die besagt, dass Sie nicht den Schutz eines ungeschützten Objekts aufheben können.
- wenn Sie den Schutz eines Bausteins mit dem falschen Passwort aufzuheben versuchen: Eine EngineeringTargetInvocationException wird mit einer Meldung ausgelöst, die besagt, dass das Passwort nicht akzeptiert wurde.
- wenn Sie einen Baustein mit einem leeren String als Passwort zu schützen versuchen: Eine EngineeringTargetInvocationException wird mit einer Meldung ausgelöst, die besagt, dass kein Passwort angegeben wurde.

7.18.12.8 Baustein löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Programmcode

Ändern Sie den folgenden Programmcode, um einen Baustein zu löschen:

```
//Runs through block group and deletes blocks
private static void DeleteBlocks(PlcSoftware plcsoftware)
{
    PlcBlockSystemGroup group = plcsoftware.BlockGroup;
    //  or BlockUserGroup group = ...;
    for (int i = group.Blocks.Count - 1; i >= 0; i--)
    {
        PlcBlock block = group.Blocks[i];
        if (block != null)
        {
            block.Delete();
        }
    }
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.18.12.9 Gruppe für Bausteine erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Ändern Sie folgenden Programmcode, um eine Gruppe für Bausteine zu erzeugen:

```
private static void CreateBlockGroup(PlcSoftware plcsoftware)
//Creates a block group
{
    PlcBlockSystemGroup systemGroup = plcsoftware.BlockGroup;
    PlcBlockUserGroupComposition groupComposition = systemGroup.Groups;
    PlcBlockUserGroup myCreatedGroup = groupComposition.Create("MySubGroupName");
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.18.12.10 Gruppe für Bausteine löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Programmcode

Um eine Gruppe für Bausteine zu löschen, ändern Sie folgenden Programmcode:

```
// Deletes user groups from PlcBlockSystemGroup or PlcBlockUserGroup
private static void DeleteBlockFolder(PlcSoftware plcSoftware)
{
    PlcBlockUserGroup group = plcSoftware.BlockGroup.Groups.Find("myGroup");
    //PlcBlockSystemGroup group = plcSoftware.BlockGroup;
    PlcBlockUserGroupComposition subgroups = group.Groups;
    PlcBlockUserGroup subgroup = subgroups.Find("myUserGroup");
    if (subgroup != null)
    {
        subgroup.Delete();
    }
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.18.12.11 Auf Attribute sämtlicher Bausteine zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Der Anwender kann die für alle Bausteine geltenden Attribute mit der Methode SetAttribute() festlegen. Die folgenden Codebeispiele basieren auf den zwei Attributen AutoNumber und Number (siehe Bausteine exportieren (Seite 501) für alle geltenden Attribute von Bausteinen).

Programmcode:

```
...
PlcBlockGroup blockFolder = YourUtilities.GetFolder();
var block = blockFolder.Blocks.Find("Block_1");
 if ((bool)block.GetAttribute("AutoNumber")==true)
   {
      block.SetAttribute("AutoNumber",false);
   }
 block.SetAttribute("Number",2);
...
```

7.18.12.12 Einen ProDiag-FB anlegen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Der Openness-Anwender kann mit der Create-Aktion der PLCBlock-Zusammensetzung mit den folgenden Parametern einen ProDiag-FB anlegen.

1. Name
2. Auto-Nummerierungsmerker

3. Nummer (ignorieren, wenn "Auto-Nummerierungsmerker" wahr ist)
4. Programmiersprache

- Wenn der Anwender die Create-Aktion mit der Programmiersprache ProDiag aufruft, wird ein neuer FB ohne IDB angelegt.
- Wenn der Anwender die Create-Aktion mit IDB von ProDiag aufruft, dann wird IDB von ProDiag angelegt.
- In jedem anderen nicht unterstützten Fall wird eine wiederherstellbare Ausnahme ausgelöst.

Programmcode: Einen ProDiag-FB anlegen

```
...
PlcBlockGroup blockFolder = plc.BlockGroup;
PlcBlockComposition blockComposition = blockFolder.Blocks;
if (blockComposition != null)
  {
  string fbName = "ProDiag_Block";
  bool isAutoNumber = true;
  int number = 1;
  var progLang = ProgrammingLanguage.ProDiag;
  FB block = blockComposition.CreateFB(fbName, isAutoNumber, number, progLang);
  string iDBName="ProDiag_IDB";
  string instanceOfName = fbName;
  InstanceDB iDbBlock = blockComposition.CreateInstanceDB(iDBName, isAutoNumber, number,
instanceOfName);
  }
...
```

Siehe auch

Auf Überwachungen und Eigenschaften des ProDiag-FB zugreifen (Seite 317)

7.18.12.13 Auf Überwachungen und Eigenschaften des ProDiag-FB zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Auf Überwachungen des User-FB zugreifen

Der Openness-Anwender kann mit dem folgenden Code-Snippet auf die Überwachungen des FB zugreifen. Jeder FB hat die Liste der Überwachungen einschließlich Classic und Plus PLCs.

Programmcode: Auf Überwachungen des ProDiag-FB zugreifen

```
…
PlcBlock iDB = plc.BlockGroup.Blocks.Find("FB_Block_DB");
string fbName = iDB.GetAttribute("InstanceOfName").ToString();
FB fb = (FB)plc.BlockGroup.Blocks.Find(fbName);
if (fb.Supervisions.Count > 0)Console.WriteLine("Contains supervisions");
else
Console.WriteLine("Does not contains supervisions");
…
```

Auf die Attribute des FB-Bausteins zugreifen

Der Openness-Anwender kann AssignedProDiagFB an InstanceDB über das Attribut AssignedProDiagFB festlegen (siehe Bausteine exportieren (Seite 501)). Der Anwender kann mit der Methode GetAttribute(), GetAttributes() and SetAttribute() auf die Attribute zugreifen. Der Anwender kann die Methode SetAttributes() nicht dazu verwenden, die Attribute für mehr als ein Attribut festzulegen. TIA Portal Openness löst eine Ausnahme bei Verwenden der Methode SetAttributes() aus.

Wenn das Attribut (im angegebenen Baustein) nicht unterstützt wird, wird eine wiederherstellbare Benutzerausnahme ausgelöst. Ist kein zugeordneter ProDiag-Baustein festgelegt, gibt GetAttribute() eine leere Zeichenkette zurück.

Programmcode: Zugeordneten ProDiag-FB und IDB abrufen und festlegen

```
...
 PlcBlockGroup blockFolder = plc.BlockGroup;
 PlcBlock instanceDB = blockFolder.Blocks.Find("IDB");
 PlcBlock plcProdiag = blockFolder.Blocks.Find("block_Prodiag");
 instanceDB.SetAttribute("AssignedProDiagFB", plcProdiag.Name);
 var assignedProDiagFB = instanceDB.GetAttribute("AssignedProDiagFB");
...
```

Siehe auch

Einen ProDiag-FB anlegen (Seite 316)

7.18.12.14 ProDiag-FB-Bausteine und -Attribute lesen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Anwendung

Sie können TIA Portal Openness verwenden, um die Version des ProDiag-Funktionsbausteins und weitere Attributwerte in Verbindung mit ProDiag zu lesen. Mit GetAttribute ( ) und GetAttributes ( ) können Sie die vorhandenen sprachspezifischen Attribute von ProDiag-FBs lesen.

Attribute

Die folgenden Attribute werden von ProDiag-FBs in Openness unterstützt:

Attribute	Typ
ProDiagVersion	Version
InitialValueAcquisition	Bool
UseCentralTimeStamp	Bool

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

7.18.12.15 Externe Datei hinzufügen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet:
  Siehe Projekt öffnen (Seite 99)

Verwendung

Sie können einem PLC eine externe Datei hinzufügen. Diese externe Datei wird im Dateisystem unter dem definierten Pfad gespeichert.

Folgende Formate werden unterstützt:

- AWL
- SCL
- DB
- UDT

Hinweis

Der Zugriff auf Gruppen im Ordner „Externe Quelldateien" wird nicht unterstützt.

Es wird eine Ausnahme ausgelöst, wenn Sie eine andere Dateierweiterung als *.AWL, *.SCL, *.DB oder *.UDT angeben.

Programmcode

Ändern Sie den folgenden Programmcode, um im Ordner "Externe Quelldateien" eine externe Datei aus einem Baustein zu erzeugen:

```
private static void CreateBlockFromFile(PlcSoftware plcSoftware)
// Creates a block from a AWL, SCL, DB or UDT file
{
    PlcExternalSource externalSource =
plcSoftware.ExternalSourceGroup.ExternalSources.CreateFromFile("SomeBlockNameHere","SomePa
thHere");
}
```

7.18.12.16 Quelle aus Baustein generieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt die Generierung von Quellen in UTF-8 aus AWL- oder SCL-Bausteinen, Datenbausteinen und PLC-Typen (Anwenderdatentypen). Um eine Quelldatei eines Bausteins zu generieren, rufen Sie die Methode `GenerateSource` auf der Instanz `PlcExternalSourceSystemGroup` auf.

Der Umfang der generierten Quelldatei ist von der Generierungsoption dieser Funktion abhängig:

- GenerateOptions.None
  Quelle nur aus bereitgestellten Bausteinen generieren.
- GenerateOptions.WithDependencies
  Quelle einschließlich aller abhängigen Objekte generieren.

Die Schnittstelle `Siemens.Engineering.SW.ExternalSources.IGenerateSource` zeigt an, dass eine Quelle generiert werden kann.

Für Bausteine werden nur die Programmiersprachen AWL und SCL unterstützt. In den folgenden Fällen werden Ausnahmen ausgelöst:

- Die Programmiersprache ist nicht AWL oder SCL
- Eine Datei gleichen Namens ist bereits am Zielspeicherort vorhanden.

Für Anwenderdatentypen wird nur die Dateierweiterung "*.udt" unterstützt. In den folgenden Fällen werden Ausnahmen ausgelöst:

- Die Dateierweiterung für DBs ist nicht "*.db".
- Die Dateierweiterung für AWL-Bausteine ist nicht "*.awl".
- Die Dateierweiterung für SCL-Bausteine ist nicht "*.scl.

Programmcode

Um Quelldateien aus Bausteinen und Typen zu erzeugen, ändern Sie folgenden Programmcode:

```
//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.
...
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.18.12.17 Bausteine aus Quelle erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Verwendung

In der Gruppe "Externe Quelldateien" können Sie Bausteine aus allen externen Dateien erzeugen. Es werden nur externe Dateien mit dem Format ASCII unterstützt.

Hinweis

Der Zugriff auf Gruppen im Ordner „Externe Quelldateien" wird nicht unterstützt.

Vorhandene Bausteine werden überschrieben.

Wenn beim Aufruf ein Fehler auftritt, wird eine Exception ausgelöst. Die ersten 256 Zeichen jeder Fehlermeldung sind in der Benachrichtigung der Exception enthalten. Das Projekt wird auf den Verarbeitungszustand vor der Ausführung der Methode `GenerateBlocksFromSource` zurückgesetzt.

Programmcode

Um in der Gruppe "Externe Quelldateien" aus allen externen Dateien Bausteine zu erzeugen, ändern Sie folgenden Programmcode:

```
// Creates a block from an external source file
PlcSoftware plcSoftware = ...;
foreach (PlcExternalSource plcExternalSource in
plcSoftware.ExternalSourceGroup.ExternalSources)
{
    plcExternalSource.GenerateBlocksFromSource();
}
```

7.18.12.18 Anwenderdatentyp löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Programmcode

Ändern Sie den folgenden Programmcode, um einen Benutzertyp zu löschen:

```
private static void DeleteUserDataType(PlcSoftware plcSoftware)
{
    PlcTypeSystemGroup typeGroup = plcSoftware.TypeGroup;
    PlcTypeComposition dataTypes = typeGroup.Types;
    PlcType dataType = dataTypes.Find("DataTypeName");
    if (dataType != null)
    {
        dataType.Delete();
    }
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.18.12.19 Externe Datei löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Sie haben über eine TIA Portal Openness-Anwendung ein Projekt geöffnet:
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Programmcode

Um in der Gruppe "Externe Quelldateien" eine externe Datei zu löschen, ändern Sie folgenden Programmcode:

Hinweis

Der Zugriff auf Gruppen in der Gruppe "Externe Quelldateien" wird nicht unterstützt.

```
// Deletes an external source file
private static void DeleteExternalSource(PlcSoftware plcSoftware)
{
    PlcExternalSource externalSource =
plcSoftware.ExternalSourceGroup.ExternalSources.Find("myExternalsource");
    externalSource.Delete();
}
```

7.18.12.20 Bausteineditor starten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine Instanz des TIA Portals ist mit Benutzeroberfläche geöffnet.

Programmcode

Um den zugehörigen Editor für eine Objektreferenz des Typs `PlcBlock` in der Instanz des TIA Portals zu starten, ändern Sie den folgenden Programmcode:

```
//Opens a block in a block editor
private static void StartBlockEditor(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    plcBlock.ShowInEditor();
}
```

Ändern Sie folgenden Programmcode, um den zugehörigen Editor für eine Objektreferenz des Typs `PlcType` in der Instanz des TIA Portals zu öffnen:

```
//Opens a udt in udt editor
private static void StartPlcTypEditor(PlcSoftware plcSoftware)
{
    PlcTypeComposition types = plcSoftware.TypeGroup.Types;
    PlcType udt = types.Find("my_udt");
    udt.ShowInEditor();
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.18.13 Technologieobjekte

7.18.13.1 Übersicht über Funktionen für Technologieobjekte

TIA Portal Openness unterstützt eine Auswahl von Technologieobjektfunktionen für definierte Aufgaben, die Sie mittels der Public API außerhalb des TIA Portal aufrufen können.

Sie erhalten die Codekomponenten, die für jede Aufgabe angepasst werden müssen.

Funktionen

Die folgenden Funktionen sind für Technologieobjekte verfügbar:

- Zusammensetzung von Technologieobjekten abfragen (Seite 329)
- Technologieobjekt erstellen (Seite 329)
- Technologieobjekt löschen (Seite 330)
- Technologieobjekt übersetzen (Seite 331)
- Technologieobjekte enumerieren (Seite 332)
- Technologieobjekt finden (Seite 333)
- Parameter eines Technologieobjekts enumerieren (Seite 334)
- Parameter eines Technologieobjekts finden (Seite 334)
- Parameter eines Technologieobjekts lesen (Seite 335)
- Parameter eines Technologieobjekts schreiben (Seite 336)

Siehe auch

Standard-Bibliotheken (Seite 41)

Einsatzmöglichkeiten (Seite 35)

TIA Portal Openness-Objektmodell (Seite 51)

7.18.13.2 Übersicht über Technologieobjekte und Versionen

Technologieobjekte

Die folgende Tabelle zeigt die in der Public API verfügbaren Technologieobjekte.

CPU	FW	Technologieobjekt	Version des Technologieobjekts
S7-1200	≥ V4.2	TO_PositioningAxis	V6.0
		TO_CommandTable	
		PID_Compact	V2.3
		PID_3Step	
		PID_Temp	V1.1

CPU	FW	Technologieobjekt	Version des Technologieobjekts
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	
		CONT_S	
		TCONT_CP	
		TCONT_S	
S7-300/400	Jede	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) Das Technologieobjekt unterstützt die folgenden Openness-Funktionen nicht: Schreiben von Parametern.

2) Das Technologieobjekt unterstützt die folgenden Openness-Funktionen nicht: Enumerieren von Parametern, Suchen von Parametern, Lesen von Parametern, Schreiben von Parametern.

Hinweis

S7-1500 Motion Control

Die Technologieobjekte TO_OutputCam, TO_CamTrack und TO_MeasuringInput der S7-1500 werden getrennt behandelt.

Weitere Informationen finden Sie im Abschnitt "S7-1500 Motion Control (Seite 345)".

7.18.13.3 Überblick der Datentypen

Die Datentypen der Parameter von Technologieobjekten im TIA Portal sind C#-Datentypen in der Public API zugeordnet.

Datentypen

Die folgende Tabelle zeigt die Zuordnung der Datentypen:

Format	Datentyp im TIA Portal	Datentyp in C#
Binärzahlen	Bool	bool
	BBool	bool
	Byte	byte
	Word	ushort
	DWord	uint
	LWord	ulong
Ganzzahlen	SInt	sbyte
	Int	short
	Dint	int
	LInt	long
	USInt	byte
	UInt	ushort
	UDint	uint
	ULInt	ulong
Gleitpunktzahlen	Real	float
	LReal	double
	Time	double
Zeichenfolgen	Char	char
	WChar	char
	String	string
	WString	string
Hardware-Datentypen	HW_*	ushort
	Block_*	ushort

* Platzhalter für Erweiterung des Gerätetyps im TIA Portal-Projekt

7.18.13.4 Zusammensetzung von Technologieobjekten abfragen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)

Programmcode

Um alle Technologieobjekte einer PLC zu erhalten, ändern Sie den folgenden Programmcode:

```
// Retrieves all technology objects of a PLC
private static void GetTechnologicalObjectsOfPLC(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBGroup technologicalObjectGroup =
plcSoftware.TechnologicalObjectGroup;
    TechnologicalInstanceDBComposition technologicalObjects =
technologicalObjectGroup.TechnologicalObjects;
}
```

Siehe auch

Standard-Bibliotheken (Seite 41)

7.18.13.5 Technologieobjekt erstellen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)

Verwendung

Nur Technologieobjekte, die im Abschnitt Übersicht über Technologieobjekte und Versionen (Seite 326) aufgeführt sind, können erstellt werden. Bei nicht unterstützten Technologieobjekten oder ungültigen Parametern wird eine Ausnahme gemeldet.

Hinweis

S7-1500 Motion Control

Die Technologieobjekte TO_OutputCam, TO_CamTrack und TO_MeasuringInput der S7-1500 werden getrennt behandelt.

Weitere Informationen finden Sie im Abschnitt "S7-1500 Motion Control (Seite 345)".

Programmcode

Um ein Technologieobjekt zu erstellen und es einer vorhandenen PLC hinzuzufügen, ändern Sie den folgenden Programmcode:

```
// Create a technology object and add to technology object composition
private static void CreateTechnologicalObject(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBComposition technologicalObjects =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects;

    string nameOfTO = "PID_Compact_1"; // How the technology object should be named
    string typeOfTO = "PID_Compact"; // How the technology object type is called, e.g. in
    // "Add new technology object"-dialog
    Version versionOfTO = new Version("2.3"); // Version of technology object
    TechnologicalInstanceDB technologicalObject = technologicalObjects.Create(nameOfTO,
typeOfTO, versionOfTO);
}
```

Mögliche Werte und Kombinationen aus Name, Typ und Version des Technologieobjekts sind im Abschnitt Übersicht über Technologieobjekte und Versionen (Seite 326) zu finden.

Siehe auch

Standard-Bibliotheken (Seite 41)

7.18.13.6 Technologieobjekt löschen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)
- Das Technologieobjekt ist vorhanden.
  Siehe Technologieobjekt finden (Seite 333)

Programmcode

Um ein Technologieobjekt zu löschen, ändern Sie den folgenden Programmcode:

```
// Delete a technology object from DB composition and from PLC
private static void DeleteTechnologicalObject(TechnologicalInstanceDB technologicalObject)
{
    technologicalObject.Delete();
}
```

Siehe auch

Standard-Bibliotheken (Seite 41)

7.18.13.7 Technologieobjekt übersetzen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)
- Das Technologieobjekt ist vorhanden.
  Siehe Technologieobjekt erstellen (Seite 329)

Programmcode: Übersetzen eines Technologieobjekts

Um ein Technologieobjekt zu übersetzen, ändern Sie den folgenden Programmcode:

```
// Compile a single technology object
private static void CompileSingleTechnologicalObject(TechnologicalInstanceDB
technologicalObject)
{
    ICompilable singleCompile = technologicalObject.GetService<ICompilable>();
    CompilerResult compileResult = singleCompile.Compile();
}
```

Programmcode: Übersetzen der Technologieobjektgruppe

Um die Technologieobjektgruppe zu übersetzen, ändern Sie den folgenden Programmcode:

```
// Compile technology object group
private static void CompileTechnologicalObjectGroup(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBGroup technologicalObjectGroup =
plcSoftware.TechnologicalObjectGroup;
    ICompilable groupCompile = technologicalObjectGroup.GetService<ICompilable>();
    CompilerResult compileResult = groupCompile.Compile();
}
```

Übersetzungsergebnisse

Die Übersetzungsergebnisse von Technologieobjekten werden rekursiv gespeichert.

Ein Beispiel für die rekursive Auswertung von Übersetzungsergebnissen finden Sie in Abschnitt "Projekt übersetzen (Seite 118)".

Siehe auch

Standard-Bibliotheken (Seite 41)

7.18.13.8 Technologieobjekte enumerieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)

Programmcode

Um Technologieobjekte zu enumerieren, ändern Sie den folgenden Programmcode:

```
// Enumerate all technology objects
private static void EnumerateTechnologicalObjects(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBComposition technologicalObjects =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects;
    foreach (TechnologicalInstanceDB technologicalObject in technologicalObjects)
    {
        // Do something ...
    }
}
```

Siehe auch

Standard-Bibliotheken (Seite 41)

7.18.13.9 Technologieobjekt finden

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)

Programmcode

Um ein spezifisches Technologieobjekt zu finden, ändern Sie den folgenden Programmcode:

```
// Find a specific technology object by its name
private static void FindTechnologicalObject(PlcSoftware plcSoftware)
{
    TechnologicalInstanceDBComposition technologicalObjects =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects;
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject = technologicalObjects.Find(nameOfTO);
}
```

Siehe auch

Standard-Bibliotheken (Seite 41)

7.18.13.10 Parameter eines Technologieobjekts enumerieren

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)
- Ein Technologieobjekt ist vorhanden.
  Siehe Technologieobjekt erstellen (Seite 329) oder Parameter eines Technologieobjekts finden (Seite 334)
- Das Technologieobjekt (Seite 326) unterstützt diese Funktion.

Programmcode

Um Parameter eines bestimmten Technologieobjekts zu enumerieren, ändern Sie den folgenden Programmcode:

```
// Enumerate parameters of a technology object
private static void EnumerateParameters(PlcSoftware plcSoftware)
{
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find(nameOfTO);

    foreach (TechnologicalParameter parameter in technologicalObject.Parameters)
    {
        // Do something ...
    }
}
```

Siehe auch

Standard-Bibliotheken (Seite 41)

Technologieobjekt finden (Seite 333)

7.18.13.11 Parameter eines Technologieobjekts finden

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)
- Ein Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329)
- Das Technologieobjekt (Seite 326) unterstützt diese Funktion.

Programmcode

Um Parameter eines bestimmten Technologieobjekts zu finden, ändern Sie den folgenden Programmcode:

```
// Find parameters of a technology object
private static void FindParameterOfTechnologicalObject(PlcSoftware plcSoftware)
{
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find(nameOfTO);

    string nameOfParameter = "Config.InputUpperLimit";
    TechnologicalParameter parameter =
technologicalObject.Parameters.Find(nameOfParameter);
}
```

Parameter verschiedener Technologieobjekte

Parameter von SIMATIC S7-1200 Motion Control (Seite 337)

Parameter von SIMATIC S7-1500 Motion Control (Seite 345)

Parameter der PID-Regelung (Seite 364)

Zählparameter (Seite 365)

Parameter von Easy Motion Control (Seite 365)

Siehe auch

Standard-Bibliotheken (Seite 41)

Technologieobjekt finden (Seite 333)

7.18.13.12 Parameter eines Technologieobjekts lesen

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)
- Ein Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329)
- Das Technologieobjekt (Seite 326) unterstützt diese Funktion.

Programmcode

Um Parameter eines bestimmten Technologieobjekts zu lesen, ändern Sie den folgenden Programmcode:

```
// Read parameters of a technology object
private static void ReadParameterOfTechnologicalObject(PlcSoftware plcSoftware)
{
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find(nameOfTO);

    string nameOfParameter = "Config.InputUpperLimit";
    TechnologicalParameter parameter =
technologicalObject.Parameters.Find(nameOfParameter);

    // Read from parameter
    string name = parameter.Name;
    object value = parameter.Value;
}
```

Siehe auch

Standard-Bibliotheken (Seite 41)

Technologieobjekt finden (Seite 333)

7.18.13.13 Parameter eines Technologieobjekts schreiben

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  SieheVerbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PLC ist im Projekt ermittelt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)
- Ein Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329)
- Das Technologieobjekt (Seite 326) unterstützt diese Funktion.

Ausnahme

Eine EngineeringException wird gemeldet, wenn:

- Sie einen neuen Wert für einen Parameter festlegen, der keinen Schreibzugriff bietet.
- Ein neuer Wert für einen Parameter einen nicht unterstützten Typ hat.

Programmcode

Um Parameter eines spezifischen Technologieobjekts zu schreiben, ändern Sie den folgenden Programmcode:

```
// Write parameters of a technology object
private static void WriteParameterOfTechnologicalObject(PlcSoftware plcSoftware)
{
    string nameOfTO = "PID_Compact_1";
    TechnologicalInstanceDB technologicalObject =
plcSoftware.TechnologicalObjectGroup.TechnologicalObjects.Find(nameOfTO);

    string nameOfParameter = "Config.InputUpperLimit";
    TechnologicalParameter parameter =
technologicalObject.Parameters.Find(nameOfParameter);

    // Write to parameter if the value is writable
    object value = 3.0;
    parameter.Value = value;
}
```

Parameter der verschiedenen Technologieobjekte

Parameter von SIMATIC S7-1200 Motion Control (Seite 337)

Parameter von SIMATIC S7-1500 Motion Control (Seite 345)

Parameter der PID-Regelung (Seite 364)

Zählparameter (Seite 365)

Parameter von Easy Motion Control (Seite 365)

Siehe auch

Standard-Bibliotheken (Seite 41)

Technologieobjekt finden (Seite 333)

7.18.13.14 S7-1200 Motion Control

Version der Openness Engineering Library ändern

Wenn Sie "Openness\PublicAPI**V14 SP1**\Siemens.Engineering.dll" mit dem TIA Portal V15 benutzen, dann funktioniert Ihre aktuelle Openness-Anwendung weiterhin.

Wenn Sie mit dem TIA Portal V15 zu "Openness\PublicAPI**V15**\Siemens.Engineering.dll" wechseln, dann müssen Sie alle Zugriffe auf Array-Variablen für S7-1200 Motion Control anpassen.

Die betroffenen Arrays für TO_PositioningAxis sind in der folgenden Tabelle aufgelistet:

Zugriff in Openness < V15	Zugriff in Openness ≥ V15
_Sensor.Sensor[1].<alle Variablen>	_Sensor[1].<alle Variablen>
ControlPanel.Input.Command.Command[1].<alle Variablen>	ControlPanel.Input.Command[1].<alle Variablen>
ControlPanel.Output.Command.Command[1].<alle Variablen>	ControlPanel.Output.Command[1].<alle Variablen>
Internal.Internal[n].<alle Variablen>	Internal[n].<alle Variablen>
Sensor.Sensor[1].<alle Variablen>	Sensor[1].<alle Variablen>
StatusSensor.StatusSensor[1].<alle Variablen>	StatusSensor[1].<alle Variablen>

Die betroffenen Arrays für TO_CommandTable sind in der folgenden Tabelle aufgelistet:

Zugriff in Openness < V15	Zugriff in Openness ≥ V15
Command.Command[n].<alle Variablen>	Command[n].<alle Variablen>

PROFIdrives mit der Hardware-Adresse verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein PROFIdrive ist in dem Projekt verfügbar und mit der S7-1200 CPU verschaltet.
- Das Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329).

Programmcode

Ändern Sie den folgenden Programmcode, um einen PROFIdrive durch eine Hardware-Adresse mit „TO_PositioningAxis“ zu verschalten.

```
//An instance of the technology object axis is already available in the program before
private static void ConnectingDrive(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to PROFIdrive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 1;

    //Set axis to drive mode
    technologicalObject.Parameters.Find("_Actor.Interface.DataConnection").Value = 0;

    //Set connection to 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;
}
```

Geber für PROFIdrives mit der Hardware-Adresse verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein PROFIdrive ist in dem Projekt verfügbar und mit der S7-1200 CPU verschaltet.
- Das Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329).

Programmcode

Ändern Sie den folgenden Programmcode, um einen Geber mit der Hardware-Adresse mit dem "TO_PositioningAxis" zu verschalten:

```
//An instance of the technology object axis is already available in the program before
private static void ConnectingEncoder(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to PROFIdrive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 1;

    //Set the encoder mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.EncoderConnection").Value =
7;

    //Set axis to use PROFINET encoder
    technologicalObject.Parameters.Find("_Sensor[1].Interface.DataConnection").Value = 0;

    //Set connection to 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;
}
```

Analogantriebe mit der Hardware-Adresse verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein Analogantrieb ist in dem Projekt verfügbar und mit der S7-1200 CPU verschaltet.
- Das Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329).

Programmcode

Ändern Sie den folgenden Programmcode, um einen Analogantrieb mit der Hardware-Adresse mit dem "TO_PositioningAxis" zu verschalten:

```
//An instance of the technology object axis is already available in the program before
private static void ConnectingEncoder(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to analog drive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 0;

    //Set axis to drive mode
    technologicalObject.Parameters.Find("_Actor.Interface.DataConnection").Value = 0;

    //Set connection to analog adress of drive
    technologicalObject.Parameters.Find("_Actor.Interface.Analog").Value = "%QW64";
}
```

Geber für Analogantriebe mit der Hardware-Adresse verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein Analogantrieb ist in dem Projekt verfügbar und mit der S7-1200 CPU verschaltet.
- Das Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329).

Programmcode

Ändern Sie den folgenden Programmcode, um einen Geber mit der Hardware-Adresse mit dem "TO_PositioningAxis" zu verschalten:

```
//An instance of the technology object axis is already available in the program before
//Connecting by High Speed Counter mode
private static void ConnectingEncoder(TechnologicalInstanceDB technologicalObject)
{
    //Set axis to analog drive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 0;

    //Set encoder for high-speed counter mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.EncoderConnection").Value =
4;
    technologicalObject.Parameters.Find("_Sensor[1].Interface.HSC.Name").Value = "HSC_1";
}

    //An instance of the technology object axis is already available in the program before
    //Connecting by PROFINET/PROFIBUS telegram
    private static void ConnectingEncoder(TechnologicalInstanceDB
    technologicalObject)
{
    //Set axis to analog drive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 0;
    //Set encoder for PROFINET/PROFIBUS mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.EncoderConnection").Value =
7;
    technologicalObject.Parameters.Find("_Sensor[1].Interface.DataConnection").Value =
"Encoder";
    technologicalObject.Parameters.Find("_Sensor[1].Interface.ProfiDriveIn").Value =
"%I68.0";
    technologicalObject.Parameters.Find("Sensor[1].Interface.Number").Value = 1;
    // 1 = Encoder1, 2 = Encoder2;
}
```

Antriebe mit einem Datenbaustein verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1200 CPU ist im Projekt angelegt.

- Ein Datenbaustein ist in dem Projekt verfügbar und auf "Nicht optimiert" eingestellt
  Bei einem PROFIdrive-Achstyp enthält der Datenbaustein eine Variable dieses Typs, z. B. PD_TEL3.
  Bei einem Analogantrieb enthält der Datenbaustein eine Variable des Datentyps Wort.
- Das Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329).

Programmcode

Ändern Sie den folgenden Programmcode, um einen PROFIdrive mit einem Datenbaustein mit dem "TO_PositioningAxis" zu verschalten.

```
//An instance of the technology object axis is already available in the program before
private static void ConfigureDrivewithDataBlock(TechnologicalInstanceDB
technologicalObject)
{
    //Set axis to PROFIdrive mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 1;

    //Set axis to data block mode
    technologicalObject.Parameters.Find("_Actor.Interface.DataConnection").Value = 1;

    //Set the tag in the data block
    technologicalObject.Parameters.Find("_Actor.Interface.DataBlock").Value =
"Data_block_1.Member_of_type_PD_TEL3";
}
```

Programmcode

Ändern Sie den folgenden Programmcode, um einen Analogantrieb mit einem Datenbaustein mit dem "TO_PositioningAxis" zu verschalten:

```
//An instance of the technology object axis is already available in the program before
//Connecting an analog drive with data block.
private static void ConfigureDrivewithDataBlock(TechnologicalInstanceDB
technologicalObject)
{
    //Set axis to analog mode
    technologicalObject.Parameters.Find("Actor.Type").Value = 0;

    //Set the tag in the data block
    technologicalObject.Parameters.Find("_Actor.Interface.Analog").Value =
"Data_block_1.Static_1";
}
```

Geber mit einem Datenbaustein verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1200 CPU ist im Projekt angelegt.
- Ein Datenbaustein ist in dem Projekt verfügbar und auf "Nicht optimiert" eingestellt
  Bei PROFIdrive enthält der Datenbaustein eine Variable dieses Typs, z. B. PD_TEL3.
- Das Technologieobjekt liegt vor.
  Siehe Technologieobjekt erstellen (Seite 329).

Programmcode

Ändern Sie den folgenden Programmcode, um einen Geber mit einem Datenbaustein zu verschalten:

```
//An instance of the technology object axis is already available in the program before
private static void ConfigureEncoderwithDataBlock(TechnologicalInstanceDB
technologicalObject)
{
    //Set axis to PROFIdrive mode depending by axis type. 1 = PROFIdrive, 0 = Analog Drive.
    technologicalObject.Parameters.Find("Actor.Type").Value = 1;

    //Set the encoder mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.EncoderConnection").Value =
7;

    //Set axis to data block mode
    technologicalObject.Parameters.Find("_Sensor[1].Interface.DataConnection").Value = 1;

    //Set the tag in the data block. For PD_TEL3 and PD_TEL4 "Encoder1" or "Encoder2".
    technologicalObject.Parameters.Find("_Sensor[1].Interface.DataBlock").Value =
"Data_block_1.Member_of_Type_PD_TEL3";
}
```

Parameter für TO_PositioningAxis und TO_CommandTable

Eine Liste mit allen verfügbaren Variablen finden Sie im Funktionshandbuch SIMATIC STEP 7 S7-1200 Motion Control im Internet (https://support.industry.siemens.com/cs/ww/de/view/109754206).

Hinweis

Im TIA Portal finden Sie in der Parametersicht der Technologieobjektkonfiguration die Spalte "Name in Openness".

7.18.13.15 S7-1500 Motion Control

TO_OutputCam, TO_CamTrack und TO_MeasuringInput erstellen und finden

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_PositioningAxis, TO_SynchronousAxis oder TO_ExternalEncoder ist im Projekt angelegt.

Anwendung

Die Technologieobjekte Nocken, Nockenspur und Messtaster sind mit einem der Technologieobjekte Positionierachse, Gleichlaufachse oder Externer Geber verschaltet. Um auf die Technologieobjekte Nocken, Nockenspur oder Messtaster zuzugreifen, verwenden Sie den Service OutputCamMeasuringInputContainer.

Programmcode: Technologieobjekte Nocken, Nockenspur und Messtaster erstellen und finden

Ändern Sie den folgenden Programmcode, um das Technologieobjekt Nocken, Nockenspur oder Messtaster zu erstellen oder zu finden.

```
/*An instance of the technology object under which the TO_OutputCam, TO_CamTrack or
TO_MeasuringInput should be created is already available in the program before*/
private static void CreateFind_OutputcamCamtrackMeasuringinput(TechnologicalInstanceDB
technologyObject)
{
    //Retrieve service OutputCamMeasuringInputContainer
    OutputCamMeasuringInputContainer container =
    technologyObject.GetService<OutputCamMeasuringInputContainer>();
    //Get access to TO_OutputCam / TO_CamTrack container
    TechnologicalInstanceDBComposition outputcamCamtrackContainer = container.OutputCams;

    //Find technology object TO_OutputCam or TO_CamTrack
    TechnologicalInstanceDB outputCam = outputcamCamtrackContainer.Find("OutputCamName");
    TechnologicalInstanceDB camTrack = outputcamCamtrackContainer.Find("CamTrackName");

    //Create new technology object TO_OutputCam or TO_CamTrack
    TechnologicalInstanceDB newOutputCam =
    outputcamCamtrackContainer.Create("NewOutputCamName", "TO_OutputCam",
    new Version(3, 0));
    TechnologicalInstanceDB newCamTrack =
    outputcamCamtrackContainer.Create("NewCamTrackName", "TO_CamTrack", new Version(3, 0));

    //Get access to TO_MeasuringInput container
    TechnologicalInstanceDBComposition measuringInputContainer = container.MeasuringInputs;

    //Find technology object TO_MeasuringInput
    TechnologicalInstanceDB measuringInput =
    measuringInputContainer.Find("MeasuringInputName");

    //Create new technology object TO_MeasuringInput
    TechnologicalInstanceDB newMeasuringInput =
    measuringInputContainer.Create("NewMeasuringInput", "TO_MeasuringInput",
    new Version(3, 0));
}
```

Parameter von S7-1500 Motion Control

Die meisten Parameter von S7-1500 Motion Control Technologieobjekten sind direkt auf Datenbausteinvariablen abgebildet, doch es gibt auch einige zusätzliche Parameter, die nicht direkt auf Datenbausteine abgebildet sind. In Openness haben die direkt abgebildeten Parameter die gleiche Reihenfolge wie in der "Datennavigation" in der Parametersicht des Technologieobjekts. Nach den direkt abgebildeten Parametern folgen die zusätzlichen Parameter in der Reihenfolge der Tabelle.

Parameter, die direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind:

Sie haben Zugriff auf alle Technologieobjekt-Datenbausteinvariablen, die im Allgemeinen beschrieben sind, außer:

- Schreibgeschützte Variablen
- Variablen des Datentyps VREF
- Variablen mit der Struktur „InternalToTrace“
- Variablen mit der Struktur „ControlPanel“

Sie können zusätzliche Informationen über die direkt abgebildeten Parameter in folgenden Anhängen finden:

- Funktionshandbuch SIMATIC S7-1500 Motion Control: https://support.industry.siemens.com/cs/ww/en/view/109749262 (https://support.industry.siemens.com/cs/ww/de/view/109749262)
- Funktionshandbuch SIMATIC S7-1500T Motion Control: https://support.industry.siemens.com/cs/ww/en/view/109749263 (https://support.industry.siemens.com/cs/ww/de/view/109749263)
- Funktionshandbuch SIMATIC S7-1500T Kinematics Functions: https://support.industry.siemens.com/cs/ww/en/view/109749264 (https://support.industry.siemens.com/cs/ww/de/view/109749264)

Einige Technologieparameter, die Datenbausteinvariablen abbilden, müssen in der PublicAPI schreibbar gemacht werden. Die zulässigen Werte sind die gleichen wie die Werte für die zugrundeliegenden Datenbausteinvariablen. Die betroffenen Parameter sind in den folgenden Tabellen aufgelistet:

Name in Openness	Datentyp	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	-
Actor.DataAdaptionOffline	bool	X	X	X	-
VirtualAxis.Mode	uint	X	X	X	-
Sensor[n].DataAdaptionOffline[1)]	bool	-	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.DataAdaptionOffline	bool	-	-	-	X
Sensor.Interface.Number	uint	-	-	-	X
Sensor.Type	int	-	-	-	X

Name in Openness	Datentyp	TO_OutputCam	TO_MeasuringInput	TO_Kinematics[2)]
Interface.LogicOperation	int	X	-	-
Parameter.MeasuringInput-Type	int	-	X	-
Kinematics.TypeOfKinematics	int	-	-	X
MotionQueue.MaxNumberOfCommands	int	-	-	X

1) S7-1500 CPU: n=1; S7-1500T CPU: 1≤n≤4

2) S7-1500T CPU

Parameter, die nicht direkt auf Technologieobjekt-Datenbausteinvariablen abgebildet sind:

Für S7-1500 Motion Control Technologieobjekte sind folgende zusätzliche Parameter verfügbar, die nicht direkt auf Datenbausteinvariablen abgebildet sind:

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness	TO_SpeedAxis	TO_PositioningAxis TO_SynchronousAxis	TO_ExternalEncoder
_Properties.MotionType	Achstyp bzw. „Technische Einheit der Position“	0: Linear 1: Rotierend	int	-	X	X
_Units.LengthUnit	Positionseinheiten	Siehe Variable Units.LengthUnit[3)]	uint	-	X	X
_Units.VelocityUnit	Geschwindigkeitseinheiten	Siehe Variable Units.VelocityUnit[3)]	uint	X	X	X
_Units.TorqueUnit	Drehmomentseinheiten	Siehe Variable Units.TorqueUnit[3)]	uint	X	X	-
_Units.ForceUnit	Krafteinheiten	Siehe Variable Units.ForceUnit[3)]	uint	-	X	-
_Actor.Interface.Telegram	Antriebstelegramm	Telegramm Nummer[4)]	uint	X	X	-
_Actor.Interface.EnableDriveOutputAddress	Adresse für Ausgang Antriebsfreigabe	PublicAPI-Objekt	SW.Tags.PlcTag	X	X	-
_Actor.Interface.DriveReadyInputAddress	Adresse für Eingang Antriebsfreigabe	PublicAPI-Objekt	SW.Tags.PlcTag	X	X	-
_Sensor[n].Interface.Telegram[5)]	Gebertelegramm	Telegramm Nummer[4)]	uint	-	X	-

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp in Openness	TO_SpeedAxis	TO_PositioningAxis TO_SynchronousAxis	TO_ExternalEncoder
_Sensor[n].ActiveHoming.DigitalInputAddress[5)]	Digitaleingang	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	-
_Sensor[n].PassiveHoming.DigitalInputAddress[5)]	Digitaleingang	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	-
_PositionLimits_HW.MinSwitchAddress	Adresse negativer Hardware-Endschalter	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	-
_PositionLimits_HW.MaxSwitchAddress	Adresse positiver Hardware-Endschalter	PublicAPI-Objekt	SW.Tags.PlcTag	-	X	-
_Sensor.Interface.Telegram	Gebertelegramm	Telegramm Nummer[4)]	uint	-	-	X
_Sensor.PassiveHoming.DigitalInputAddress	Digitaleingang	PublicAPI-Objekt	SW.Tags.PlcTag	-	-	X

Für die Technologieobjekte Nocken, Nockenspur und Messtaster ist der folgende zusätzliche Parameter verfügbar:

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp
_AssociatedObject	Zugeordnete Achse oder Externer Geber	PublicAPI-Objekt	SW.TechnologicalObjects.TechnologicalInstanceDB

Beim Technologieobjekt Kinematik sind die folgenden zusätzlichen Parameter verfügbar (S7-1500T):

Name in Openness	Name in Funktionssicht	Möglicher Wert	Datentyp
_KinematicsAxis[1...4]	Achse 1 - 3, Ausrichtungsachse	Achse, die mit TO_Kinematics-Objekten verschaltet werden kann	SW.TechnologicalObjects.TechnologicalInstanceDB
_Units.LengthUnit	Maßeinheit > Position	Siehe Variable Units.LengthUnit[3)]	uint
_Units.LengthVelocityUnit	Maßeinheit > Geschwindigkeit	Siehe Variable Units.LengthVelocityUnit[3)]	uint
_Units.AngleUnit	Maßeinheit > Winkel	Siehe Variable UnitsAngleUnit[3)]	uint
_Units.AngleVelocityUnit	Maßeinheit > Winkelgeschwindigkeit	Siehe Variable Units.AngleVelocityUnit[3)]	uint

3) Mögliche Werte sind im Funktionshandbuch S7-1500 Motion Control im Kapitel Variablen Units (TO) beschrieben

4) Mögliche Werte sind im Funktionshandbuch S7-1500 Motion Control im Kapitel PROFIdrive Telegramme beschrieben

5) S7-1500 CPU: n=1; S7-1500T CPU: 1≤n≤4

Programmcode: Direkt abgebildete Datenbausteinvariablen

Ändern Sie den folgenden Programmcode, um auf die direkt abgebildeten Parameter zuzugreifen:

```
//An instance of the technology object is already available in the program before
private static void ReadWriteDataBlockTag(TechnologicalInstanceDB technologyObject)
{
    //Read value from data block tag "ReferenceSpeed"
    double value =
    (double)technologyObject.Parameters.Find("Actor.DriveParameter.ReferenceSpeed").Value;

    //Write data block tag "ReferenceSpeed"
    technologyObject.Parameters.Find("Actor.DriveParameter.ReferenceSpeed").Value = 3000.0;
}
```

Programmcode: Weitere Parameter

Ändern Sie den folgenden Programmcode, um auf die zusätzlichen Parameter zuzugreifen:

```
//An instance of the technology object is already available in the program before
private static void ReadWriteAdditionalParameter(TechnologicalInstanceDB technologyObject)
{
    //Read additional parameter "_Properties.MotionType"
    uint value = (uint)technologyObject.Parameters.Find("_Properties.MotionType").Value;

    //Write additional parameter "_Properties.MotionType"
    technologyObject.Parameters.Find("_Properties.MotionType").Value = 1;
}
```

Weitere Informationen

Weitere Informationen finden Sie hier:

- Funktionshandbuch SIMATIC S7-1500 Motion Control: https://support.industry.siemens.com/cs/ww/en/view/109749262 (https://support.industry.siemens.com/cs/ww/de/view/109749262)
- Funktionshandbuch SIMATIC S7-1500T Motion Control: https://support.industry.siemens.com/cs/ww/en/view/109749263 (https://support.industry.siemens.com/cs/ww/de/view/109749263)
- Funktionshandbuch SIMATIC S7-1500T Kinematics Functions: https://support.industry.siemens.com/cs/ww/en/view/109749264 (https://support.industry.siemens.com/cs/ww/de/view/109749264)

Antriebe verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_SpeedAxis, TO_PositioningAxis oder TO_SynchronousAxis ist im Projekt angelegt.
- Ein Antrieb ist im Projekt angelegt.

Anwendung

Um eine Achse mit einem Antrieb zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API AxisEncoderHardwareConnectionInterface bietet die folgenden Methoden, die zur Verschaltung oder Trennung der Aktor- oder Sensorschnittstellen verwendet werden können:

Methode	Beschreibung
void Connect(HW.DeviceItem moduleInOut)	Verschaltet mit Eingangs- und Ausgangsadressen an einem Modul
void Connect(HW.DeviceItem moduleIn, HW.DeviceItem moduleOut)	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen
void Connect(HW.DeviceItem moduleIn, HW.DeviceItem moduleOut, ConnectOption connectOption)	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen unter Angabe einer zusätzlichen ConnectOption
void Connect(HW.Channel channel)	Verschaltet mit einem Kanal
void Connect(int addressIn, int addressOut, ConnectOption connectOption)	Verschaltet Bit-Adressen direkt
void Connect(string pathToDBMember)	Verschaltet mit einer Datenbausteinvariablen
void Connect(SW.Tags.PlcTag outputTag)	Verschaltet mit einer CPU-Variablen
void Disconnect()	Trennt eine existierende Verschaltung

Hinweis

Automatische Verschaltung

Bitte beachten Sie, dass hier das gleiche Verhalten wie bei der Anwenderschnittstelle angewendet wird. Immer wenn die Aktorschnittstelle über eine der folgenden Verschaltungsmethoden verschaltet wird und das Telegramm ein Sensorteil oder Telegramm 750 enthält, werden diese Teile automatisch verschaltet.

Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird. Die entsprechenden Verschaltungswerte werden nur eingestellt, wenn eine Verschaltung dieser spezifischen Art besteht.

Attribut	Datentyp	Beschreibung
IsConnected	bool	TRUE: Schnittstelle ist verschaltet FALSE: Schnittstelle ist nicht verschaltet
InputOutputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangs- und Ausgangsadressen enthält
InputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangsadressen enthält Der Wert wird auch im Falle einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
OutputModule	HW.DeviceItem	Verschaltetes Modul, das Ausgangsadressen enthält Der Wert wird auch im Falle einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
InputAddress	int	Logische Eingangsadresse des verschalteten Objekts; zum Beispiel 256.
OutputAddress	int	Logische Ausgangsadresse des verschalteten Objekts; zum Beispiel 256.
ConnectOption	ConnectOption	Wert der bei der Verbindungsherstellung eingestellten ConnectOption: • Standardeinstellung Es können nur Module ausgewählt werden, die als gültige Verbindungspartner erkannt werden. • AllowAllModules Entspricht dem Markieren von "Zeige alle Module" in der Anwenderschnittstelle.
Channel	HW.Channel	Verschalteter Kanal
PathToDBMember	string	Verschaltete Technologieobjekt-Datenbausteinvariable
OutputTag	SW.Tags.PlcTag	Verschaltete CPU-Variable (analoge Verschaltung)
SensorIndexInActor-Telegram	int	Verschalteter Sensorteil in Aktortelegramm Das Attribut ist nur für Sensorschnittstellen relevant. 0: Geber ist nicht verschaltet 1: Geber ist mit der ersten Sensorschnittstelle im Telegramm verschaltet 2: Geber ist mit der zweiten Sensorschnittstelle im Telegramm verschaltet Der Wert der Aktorschnittstelle ist stets 0.

Hinweis

Zugriff auf die Sensorschnittstelle

Um auf die Sensorschnittstelle zuzugreifen, können Sie SensorInterface[m] mit 0≤m≤3 verwenden.

Programmcode: void Connect(HW.DeviceItem moduleInOut)

Ändern Sie den folgenden Programmcode, um ein gemischtes Modul, das Eingangs- und Ausgangsadressen enthält, zu verschalten.

```
//An instance of technology object and device item is already available in the program
before
private static void 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;
}
```

Telegramm 750 verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_SpeedAxis, TO_PositioningAxis oder TO_SynchronousAxis V4.0 ist im Projekt angelegt.
- Ein Antrieb, der Telegramm 750 unterstützt, ist im Projekt angelegt.

Anwendung

Wenn Telegramm 750 nach Verschalten des Antriebs und der Achse hinzugefügt wurde, ist es nötig, Telegramm 750 getrennt zu verschalten. EnableTorqueData ist automatisch auf TRUE gesetzt. Der Typ der Public API TorqueHardwareConnectionInterface bietet die folgenden Methoden, die zur Verschaltung oder Trennung von Telegramm 750 verwendet werden können:

Methode	Beschreibung
void Connect(HW.DeviceItem moduleInOut)	Verschaltet mit Eingangs- und Ausgangsadressen an einem Modul
void Connect(HW.DeviceItem moduleIn, HW.DeviceItem moduleOut)	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen
void Connect(HW.DeviceItem moduleIn, HW.DeviceItem moduleOut, ConnectOption connectOption)	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen unter Angabe einer zusätzlichen ConnectOption
void Connect(int addressIn, int addressOut, ConnectOption connectOption)	Verschaltet Bit-Adressen direkt
void Connect(string pathToDBMember)	Verschaltet mit einer Datenbausteinvariablen
void Disconnect()	Trennt eine existierende Verschaltung

Die TorqueHardwareConnectionInterface kann über die Eigenschaft TorqueInterface am Typ AxisHardwareConnectionProvider abgerufen werden. Wird die Verschaltung mit Telegramm 750 nicht unterstützt, ist der Wert der Eigenschaft "null".

Ist der Antrieb über Datenbausteinvariablen verschaltet, können Sie Telegramm 750 nicht mittels Modul verschalten. Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird. Die entsprechenden Verschaltungswerte werden nur eingestellt, wenn eine Verschaltung dieser spezifischen Art besteht:

Attribut	Datentyp	Beschreibung
IsConnected	bool	TRUE: Schnittstelle ist verschaltet FALSE: Schnittstelle ist nicht verschaltet
InputOutputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangs- und Ausgangsadressen enthält
InputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangsadressen enthält. Der Wert wird auch im Fall einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
OutputModule	HW.DeviceItem	Verschaltetes Modul, das Ausgangsadressen enthält. Der Wert wird auch im Fall einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
InputAddress	int	Logische Eingangsadresse des verschalteten Objekts, beispielsweise 256

Attribut	Datentyp	Beschreibung
OutputAddress	int	Logische Ausgangsadresse des verschalteten Objekts, beispielsweise 256
ConnectOption	ConnectOption	Wert der bei der Verschaltungsherstellung eingestellten ConnectOption: • Standard Es können nur Module ausgewählt werden, die als gültige Verschaltungspartner erkannt werden. • AllowAllModules Entspricht dem Auswählen von "Zeige alle Module" in der Anwenderschnittstelle.
PathToDBMember	string	Verschaltete Technologieobjekt-Datenbausteinvariable

Programmcode: Telegramm 750 verschalten

Ändern Sie den folgenden Programmcode, um ein gemischtes Modul, das Eingangs- und Ausgangsadressen enthält, zu verschalten:

```
//An instance of technology object and device item is already available in the program
before
private static void ConnectTorqueInterface(TechnologicalInstanceDB technologyObject,
DeviceItem devItem)
{
    //Retrieve service AxisHardwareConnectionProvider
    AxisHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<AxisHardwareConnectionProvider>();
    //Connect TorqueInterface with DeviceItem
    connectionProvider.TorqueInterface.Connect(devItem);
}
```

Geber verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_ExternalEncoder ist im Projekt angelegt.
- Im Projekt wird ein Objekt bestimmt, welches PROFIdrive Telegramm 81 oder 83 bereitstellt.

Anwendung

Um ein Technologieobjekt Externer Geber mit der Geber-Hardware zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API AxisEncoderHardwareConnectionInterface bietet die folgenden Methoden, die zur Verschaltung oder Trennung der Sensorschnittstelle verwendet werden können.

Methode	Beschreibung
void Connect(HW.DeviceItem moduleInOut)	Verschaltet mit Eingangs- und Ausgangsadressen an einem Modul
void Connect(HW.DeviceItem moduleIn, HW.DeviceItem moduleOut)	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen
void Connect(HW.DeviceItem moduleIn, HW.DeviceItem moduleOut, ConnectOption connectOption)	Verschaltet mit Eingangs- und Ausgangsadressen an getrennten Modulen unter Angabe einer zusätzlichen ConnectOption
void Connect(HW.Channel channel)	Verschaltet mit einem Kanal
void Connect(int addressIn, int addressOut, ConnectOption connectOption)	Verschaltet Bit-Adressen direkt
void Connect(string pathToDBMember)	Verschaltet mit einer Datenbausteinvariablen
void Connect(SW.Tags.PlcTag outputTag)	Nicht relevant für das Verschalten von Gebern
void Disconnect()	Trennt eine existierende Verschaltung

Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird. Die entsprechenden Verschaltungswerte werden nur eingestellt, wenn eine Verschaltung dieser spezifischen Art besteht.

Attribut	Datentyp	Beschreibung
IsConnected	bool	TRUE: Schnittstelle ist verschaltet FALSE: Schnittstelle ist nicht verschaltet
InputOutputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangs- und Ausgangsadressen enthält
InputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangsadressen enthält Der Wert wird auch im Falle einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
OutputModule	HW.DeviceItem	Verschaltetes Modul, das Ausgangsadressen enthält Der Wert wird auch im Falle einer bestehenden Verschaltung zu einem Modul, das Eingangs- und Ausgangsadressen enthält, eingestellt.
InputAddress	int	Die logische Eingangsadresse des verschalteten Objekts ist beispielsweise 256.
OutputAddress	int	Die logische Ausgangsadresse des verschalteten Objekts ist beispielsweise 256.
ConnectOption	ConnectOption	Wert der bei der Verbindungsherstellung eingestellten ConnectOption: • Default Es können nur Module ausgewählt werden, die als gültige Verbindungspartner erkannt werden. • AllowAllModules Entspricht dem Markieren von "Zeige alle Module" in der Anwenderschnittstelle.
Channel	HW.Channel	Verschalteter Kanal
PathToDBMember	string	Verschaltete Datenbausteinvariable

Attribut	Datentyp	Beschreibung
OutputTag	SW.Tags.PlcTag	Nicht relevant für das Verschalten von Gebern
SensorIndexInActor-Telegram	int	Verschaltetes Sensortelegramm Das Attribut ist nur für Sensorschnittstellen relevant. 0: Geber ist nicht verschaltet 1: Geber ist mit der ersten Sensorschnittstelle im Telegramm verschaltet 2: Geber ist mit der zweiten Sensorschnittstelle im Telegramm verschaltet Der Wert der Aktorschnittstelle ist stets 0.

Programmcode: Geber verschalten

Ändern Sie den folgenden Programmcode, um ein Technologieobjekt Externer Geber zu verschalten:

```
//An instance of technology object and device item is already available in the program
before
private static void UseServiceEncoderHardwareConnectionProvider(TechnologicalInstanceDB
technologyObject, DeviceItem devItem)
{
    //Retrieve service EncoderHardwareConnectionProvider
    EncoderHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<EncoderHardwareConnectionProvider>();

    //Connect SensorInterface with DeviceItem
    connectionProvider.SensorInterface.Connect(devItem);

    //Check ConnectionState of SensorInterface
    bool sensorInterfaceConnectionState = connectionProvider.SensorInterface.IsConnected;
}
```

Nocken und Nockenspur mit Hardware verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_OutputCam oder TO_CamTrack ist im Projekt angelegt.
- Ein Digitalausgabemodul ist im Projekt angelegt, zum Beispiel TM Timer DIDQ.

Anwendung

Um ein Technologieobjekt Nocken oder Nockenspur mit einem Digitalausgang zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API OutputCamHardwareConnectionProvider bietet die folgenden Methoden, die zur Verschaltung oder Trennung der Aktor- oder Sensorschnittstellen verwendet werden können:

Methode	Beschreibung
void Connect(HW.Channel channel)	Verschaltet mit einem Kanal
void Connect(SW.Tags.PlcTag outputTag)	Verschaltet mit einer CPU-Variablen
void Connect(int address)	Verschaltet Bit-Adressen direkt
void Disconnect()	Trennt eine existierende Verschaltung

Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird:

Attribut	Datentyp	Beschreibung
IsConnected	bool	TRUE: Technologieobjekt ist verschaltet FALSE: Technologieobjekt ist nicht verschaltet
Channel	HW.Channel	Verschalteter Kanal
OutputTag	SW.Tags.PlcTag	Verschaltete CPU-Variable
OutputAddress	int	Die logische Ausgangsadresse des verschalteten Objekts ist beispielsweise 256.

Programmcode: Technologieobjekt Nocken oder Nockenspur verschalten

Ändern Sie den folgenden Programmcode, um ein Technologieobjekt Nocken oder Nockenspur zu verschalten:

```
//An instance of technology object and channel item is already available in the program
before
private static void UseServiceOutputCamHardwareConnectionProvider(TechnologicalInstanceDB
technologyObject, Channel channel)

{
    //Retrieve service OutputCamHardwareConnectionProvider
    OutputCamHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<OutputCamHardwareConnectionProvider>();

    //Connect technology object with Channel
    connectionProvider.Connect(channel);

    //Check ConnectionState of technology object
    bool connectionState = connectionProvider.IsConnected;
}
```

Messtaster mit Hardware verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_MeasuringInput ist im Projekt angelegt.
- Ein Digitaleingabemodul ist am Antrieb oder im Projekt angelegt, zum Beispiel TM Timer DIDQ.

Anwendung

Um ein Technologieobjekt Messtaster mit einem Digitaleingang zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API MeasuringInputHardwareConnectionProvider bietet die folgenden Methoden, die zur Verschaltung oder Trennung der Aktor- oder Sensorschnittstellen verwendet werden können:

Methode	Beschreibung
void Connect(HW.Channel channel)	Verschaltet mit einem Kanal
void Connect(HW.DeviceItem moduleIn, int channelIndex)	Verschaltet mit einem Modul und legt einen zusätzlichen Kanalindex fest
void Connect(int address)	Verschaltet Bit-Adressen direkt
void Disconnect()	Trennt eine existierende Verschaltung

Sie können die folgenden schreibgeschützten Attribute verwenden, um festzustellen, wie das Technologieobjekt verschaltet wird:

Attribut	Datentyp	Beschreibung
IsConnected	bool	TRUE: Technologieobjekt ist verschaltet FALSE: Technologieobjekt ist nicht verschaltet
InputModule	HW.DeviceItem	Verschaltetes Modul, das Eingangsadressen enthält
ChannelIndex	int	Index des verschalteten Kanals hinsichtlich des Eingabemoduls
Channel	HW.Channel	Verschalteter Kanal
InputAddress	int	Die logische Eingangsadresse des verschalteten Objekts ist beispielsweise 256.

Programmcode: Technologieobjekt Messtaster verschalten

Ändern Sie den folgenden Programmcode, um ein Technologieobjekt Messtaster zu verschalten:

```
//An instance of technology object and channel item is already available in the program
before
private static void
UseServiceMeasuringInputHardwareConnectionProvider(TechnologicalInstanceDB
technologyObject, Channel channel)
{
    //Retrieve service MeasuringInputHardwareConnectionProvider
    MeasuringInputHardwareConnectionProvider connectionProvider =
    technologyObject.GetService<MeasuringInputHardwareConnectionProvider>();

    //Connect technology object with Channel
    connectionProvider.Connect(channel);

    //Check ConnectionState of technology object
    bool connectionState = connectionProvider.IsConnected;
}
```

Gleichlaufachse mit Leitwerten verschalten

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1500 CPU ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_PositioningAxis, TO_SynchronousAxis oder TO_ExternalEncoder als Leitachse ist im Projekt angelegt.
- Ein Technologieobjekt des Typs TO_SynchronousAxis als Folgeachse ist im Projekt angelegt.

Anwendung

Um ein Technologieobjekt Gleichlaufachse mit Leitwerten zu verschalten, ist es erforderlich, mehrere Werte zusammen in einem einzigen Aufruf anzugeben. Der Typ der Public API SynchronousAxisMasterValues bietet die folgenden Methoden, die zur Verschaltung oder Trennung von Leitwerten verwendet werden können. Leitwerte können als Sollwertkopplungen (S7-1500 CPU, S7-1500T CPU) oder Istwertkopplungen (S7-1500T CPU) verschaltet werden. Alle Methoden und Attribute sind für beide Arten von Kopplungen relevant.

Methode	Beschreibung
int IndexOf (TechnologicalInstanceDB element)	Gibt den entsprechenden Index eines Leitwertes zurück
bool Contains (TechnologicalInstanceDB element)	TRUE: Der Behälter enthält den Leitwert FALSE: Der Behälter enthält nicht den Leitwert
IEnumerator GetEnumerator <TechnologicalInstanceDB>()	Dient zur Unterstützung jeder Iteration
void Add (TechnologicalInstanceDB element)	Verschaltet die Folgeachse mit einem Leitwert
bool Remove (TechnologicalInstanceDB element)	Trennt die Folgeachse vom Leitwert TRUE: Verschaltung wurde erfolgreich getrennt FALSE: Verschaltung konnte nicht getrennt werden

Sie können die folgenden schreibgeschützten Attribute verwenden:

Attribut	Datentyp	Beschreibung
Count	int	Anzahl der Leitwerte
IsReadonly	bool	TRUE: Der Behälter ist schreibgeschützt FALSE: Der Behälter ist nicht schreibgeschützt
Parent	IEngineeringObject	Gibt die übergeordneten Werte des Behälters zurück. In diesem Fall ist der übergeordnete Wert der Service SynchronousAxis-MasterValues.
this [id] { get; }	TechnologicalInstan-ceDB	Indexbasierter Zugriff auf Leitwerte

Programmcode: Gleichlaufachse mit einem Leitwert verschalten

Ändern Sie den folgenden Programmcode, um eine Gleichlaufachse mit einem Leitwert zu verschalten:

```
//An instance of leading axis and following axis is already available in the program before
private static void UseServiceSynchronousAxisMasterValues(TechnologicalInstanceDB
masterTechnologyObject, TechnologicalInstanceDB synchronousTechnologyObject)
{
    //Retrieve service SynchronousAxisMasterValues
    SynchronousAxisMasterValues masterValues =
    synchronousTechnologyObject.GetService<SynchronousAxisMasterValues>();

    //Connect following axis and leading axis with setpoint coupling
    masterValues.SetPointCoupling.Add(masterTechnologyObject);

    //Get container of connected leading axis with setpoint coupling
    TechnologicalInstanceDBAssociation setPointMasterValues =
    masterValues.SetPointCoupling;

    //Remove connected leading axis with setpoint coupling
    masterValues.SetPointCoupling.Remove(masterTechnologyObject);

    //Connect following axis and leading axis with actual value coupling
    masterValues.ActualValueCoupling.Add(masterTechnologyObject);

    //Get container of connected leading axis with actual value coupling
    TechnologicalInstanceDBAssociation actualValueMasterValues =
    masterValues.ActualValueCoupling;

    //Remove connected leading axis with actual value coupling
    masterValues.ActualValueCoupling.Remove(masterTechnologyObject);
}
```

Exportieren und Importieren des Technologieobjekts Cam (S7-1500T)

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74).
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99).
- Eine S7-1500 CPU ist im Projekt angelegt.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)
- Das Technologieobjekt liegt vor.

Anwendung

Um die Daten eines Technologieobjekts Cam zu exportieren oder zu importieren, müssen Sie das Format und das gewünschte Trennzeichen angeben. Der Typ der Public API CamDataSupport bietet die folgenden Methoden, die zur Exportieren der Daten des Technologieobjekts Cam verwendet werden können:

Methode	Beschreibung
void SaveCamDataBinary(System.IO.FileInfo destinationFile)	Exportiert die Daten im Binärformat in die Zieldatei.
void SaveCamDataPointList(System.IO.FileInfo destinationFile, CamDataFormatSeparator separator, int samplePoints)	Exportiert die Daten im Format "PointList" in die Zieldatei.
void SaveCamData(System.IO.FileInfo destinationFile, CamDataFormat format, CamDataFormatSeparator separator)	Exportiert die Daten in die Zieldatei. Sie können das Datenformat als “MCD”, “SCOUT” oder “Pointlist” und als Trennzeichen “Tabulator” oder “Komma” angeben. Wenn Sie “PointList” wählen, werden 360 Interpolationspunkte exportiert.
void LoadCamData(System.IO.FileInfo sourceFile, CamDataFormatSeparator separator)	Importiert die Cam-Daten im Format “MCD”, “SCOUT” oder “Pointlist” in das Projekt.
void LoadCamDataBinary(System.IO.FileInfo sourceFile)	Importiert die Cam-Daten aus einer Binärdatei in das Projekt.

Die folgenden Attribute können Sie verwenden:

Attribut	Datentyp	Beschreibung
separator	CamDataFormatSeparator	Zulässige Werte • Tabulator • Komma
samplePoints	int	Anzahl zu exportierender Interpolationspunkte.
format	CamDataFormat	Zulässige Werte • MCD • SCOUT • Pointlist
destinationFile	System.IO.FileInfo	Name der Zieldatei. Darf nicht null sein. Zugriffsrechte und ausreichend Speicherplatz müssen auf dem Speichermedium gegeben sein. Eine vorhandene Datei wird überschrieben.
sourceFile	System.IO.FileInfo	Name der Quelldatei. Darf nicht null sein. Zugriffsrechte müssen gegeben sein. Der Inhalt muss im angegebenen Format vorliegen.

Programmcode: Cam-Daten exportieren

Ändern Sie folgenden Programmcode, um Cam-Daten zu exportieren:

```
//An instance of technology object is already available in the program before
private static void ExportCamData(TechnologicalInstanceDB technologyObject,
System.IO.FileInfo destinationFile)
{
    //Retrieve service CamDataSupport
    CamDataSupport camData = technologyObject.GetService<CamDataSupport>();

    //Save cam data in MCD format, using the separator Tab
    camData.SaveCamData(destinationFile, CamDataFormat.MCD, CamDataFormatSeparator.Tab);
}
```

Programmcode: Cam-Daten importieren

Ändern Sie folgenden Programmcode, um Cam-Daten zu importieren:

```
//An instance of technology object is already available in the program before
private static void ImportCamData(TechnologicalInstanceDB technologyObject,
System.IO.FileInfo sourceFile)
{
    //Retrieve service CamDataSupport
    CamDataSupport camData = technologyObject.GetService<CamDataSupport>();

    //Load cam data from source file, using the separator Tab
    camData.LoadCamData(sourceFile, CamDataFormatSeparator.Tab);
}
```

7.18.13.16 PID-Regelung

Parameter für PID_Compact, PID_3Step, PID_Temp, CONT_C, CONT_S, TCONT_CP und TCONT_S

Eine Liste sämtlicher verfügbarer Parameter finden Sie in der Produktinformation "Parameter von Technologieobjekten in TIA Portal Openness" im Internet (https://support.industry.siemens.com/cs/ww/de/view/109744932).

Für jeden Parameter werden die folgenden Eigenschaften angegeben:

- Name in der Konfiguration (TIA Portal)
- Name in Openness
- Datentyp in Openness
- Standardzugriff
- Wertebereich

Hinweis

Im TIA Portal finden Sie in der Parametersicht der Technologieobjektkonfiguration die Spalte "Name in Openness".

Weitere Informationen

Weitere Informationen finden Sie im Funktionshandbuch SIMATIC S7-1200/S7-1500 PID control im Internet (https://support.industry.siemens.com/cs/ww/de/view/108210036).

7.18.13.17 Zählen

Parameter für High_Speed_Counter und SSI_Absolute_Encoder

Eine Liste sämtlicher verfügbarer Parameter finden Sie in der Produktinformation "Parameter von Technologieobjekten in TIA Portal Openness" im Internet (https://support.industry.siemens.com/cs/ww/de/view/109744932).

Für jeden Parameter werden die folgenden Eigenschaften angegeben:

- Name in der Konfiguration (TIA Portal)
- Name in Openness
- Datentyp in Openness
- Standardzugriff
- Wertebereich

Weitere Informationen

Weitere Informationen finden Sie im Funktionshandbuch SIMATIC S7-1500, ET 200MP, ET 200SP Zählen, Messen und Positionserfassung im Internet (http://support.automation.siemens.com/WW/view/de/59709820).

7.18.13.18 Easy Motion Control

Parameter für AXIS_REF

Eine Liste sämtlicher verfügbarer Parameter finden Sie in der Produktinformation "Parameter von Technologieobjekten in TIA Portal Openness" im Internet (https://support.industry.siemens.com/cs/ww/de/view/109744932).

Für jeden Parameter werden die folgenden Eigenschaften angegeben:

- Name in der Konfiguration (TIA Portal)
- Name in Openness
- Datentyp in Openness

- Standardzugriff
- Wertebereich

Hinweis

Im TIA Portal finden Sie in der Parametersicht der Technologieobjektkonfiguration die Spalte "Name in Openness".

Weitere Informationen

Weitere Informationen zu Easy Motion Control finden Sie im Informationssystem von STEP 7 (TIA Portal).

7.18.14 Variablen und Variablentabellen

7.18.14.1 Starten des PLC-Variableneditors

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine Instanz des TIA Portals ist mit Benutzeroberfläche geöffnet.

Programmcode

Ändern Sie den folgenden Programmcode, um den entsprechenden Editor für eine Objektreferenz des Typs `PlcTagTable` in der Instanz des TIA Portals zu starten:

```
//Öffnet Variablentabelle im Editor "Tags"
private static void OpenTagtableInEditor(PlcSoftware plcSoftware)
{
    PlcTagTable plcTagTable = plcSoftware.TagTableGroup.TagTables.Find("MyTagTable");
plcTagTable.ShowInEditor();
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

7.18.14.2 Systemgruppen für PLC-Variablen abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PlcSoftware-Instanz wurde von einem PLC-Geräteelement abgerufen.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)

Programmcode

Ändern Sie den folgenden Programmcode, um eine Systemgruppe für PLC-Variablen abzufragen:

```
//Retrieves the plc tag table group from a plc
private PlcTagTableSystemGroup GetControllerTagfolder(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    return plcTagTableSystemGroup;
}
```

7.18.14.3 PLC-Variablentabelle anlegen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PlcSoftware-Instanz wurde von einem PLC-Geräteelement abgerufen.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)

Programmcode

Um die PLC-Variablentabelle anzulegen, ändern Sie folgenden Programmcode: Es wird eine neue Variablentabelle mit dem angegebenen Namen in der Zusammensetzung angelegt.

```
PlcTagTable myTable = plc.TagTableGroup.TagTables.Create("myTable");
```

Siehe auch

PLC-Target und HMI-Target abfragen (Seite 169)

7.18.14.4 Benutzerdefinierte Gruppen für PLC-Variablen enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Eine PlcSoftware-Instanz wurde von einem PLC-Geräteelement abgerufen.
  Siehe PLC-Target und HMI-Target abfragen (Seite 169)

Verwendung

Enthaltene Unterordner werden beim Enumerieren rekursiv berücksichtigt.

Programmcode: Benutzerdefinierte Gruppen für PLC-Variablen enumerieren

Ändern Sie den folgenden Programmcode, um benutzerdefinierte Gruppen für PLC-Variablen zu enumerieren:

```
//Enumerates all plc tag table user groups including subgroups
private static void EnumeratePlcTagTableUserGroups(PlcSoftware plcSoftware)
{
    foreach (PlcTagTableUserGroup plcTagTableUsergroup in plcSoftware.TagTableGroup.Groups)
    {
        EnumerateTagTableUserGroups(plcTagTableUsergroup);
    }
}
private static void EnumerateTagTableUserGroups(PlcTagTableUserGroup tagTableUsergroup)
{
    foreach (PlcTagTableUserGroup plcTagTableUsergroup in tagTableUsergroup.Groups)
    {
        EnumerateTagTableUserGroups(plcTagTableUsergroup);
        // recursion
    }
}
```

Programmcode: Auf eine benutzerdefinierte Gruppe zugreifen

Ändern Sie den folgenden Programmcode, um auf eine benutzerdefinierte Gruppe für PLC-Variablen zuzugreifen:

```
//Gives individual access to a specific plc tag table user folder
private static void AccessPlcTagTableUserGroupWithFind(PlcSoftware plcSoftware, string
folderToFind)
{
    PlcTagTableUserGroupComposition plcTagTableUserGroupComposition =
plcSoftware.TagTableGroup.Groups;
    PlcTagTableUserGroup controllerTagUserFolder =
plcTagTableUserGroupComposition.Find(folderToFind);
    // The parameter specifies the name of the user folder
}
```

7.18.14.5 Benutzerdefinierte Gruppen für PLC-Variablen erzeugen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt die Erzeugung einer benutzerdefinierten Gruppe für PLC-Variablen.

Programmcode

Ändern Sie den folgenden Programmcode, um eine benutzerdefinierte Gruppe für PLC-Variablen zu erzeugen:

```
//Creates a plc tag table user group
private static void CreatePlcTagTableUserGroup(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup systemGroup = plcSoftware.TagTableGroup;
    PlcTagTableUserGroupComposition groupComposition = systemGroup.Groups;
    PlcTagTableUserGroup myCreatedGroup = groupComposition.Create("MySubGroupName");
    // Optional;
    // create a subgroup
    PlcTagTableUserGroup mySubCreatedGroup =
myCreatedGroup.Groups.Create("MySubSubGroupName");
}
```

7.18.14.6 Benutzerdefinierte Gruppen für PLC-Variablen löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt das Löschen einer bestimmten benutzerdefinierten Gruppe für PLC-Variablentabellen.

Programmcode

Ändern Sie den folgenden Programmcode, um eine bestimmte benutzerdefinierte Gruppe für PLC-Variablentabellen zu löschen:

```
private static void DeletePlcTagTableUserGroup(PlcSoftware plcSoftware)
{
    PlcTagTableUserGroup group = plcSoftware.TagTableGroup.Groups.Find("MySubGroupName");
    if (group != null)
    {
        group.Delete();
    }
}
```

7.18.14.7 PLC-Variablentabellen in einem Ordner enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode: PLC-Variablentabellen enumerieren

Ändern Sie den folgenden Programmcode, um alle PLC-Variablentabellen in Systemgruppen oder benutzerdefinierten Gruppen zu enumerieren:

```
//Enumerates all plc tag tables in a specific system group or and user group
private static void EnumerateAllPlcTagTablesInFolder(PlcSoftware plcSoftware)
{
    PlcTagTableComposition tagTables = plcSoftware.TagTableGroup.TagTables;
    // alternatively, PlcTagTableComposition tagTables =
plcSoftware.TagTableGroup.Groups.Find("UserGroup XYZ").TagTables;
    foreach (PlcTagTable tagTable in tagTables)
    {
        // add code here
    }
}
```

Programmcode: Auf PLC-Variablentabellen zugreifen

Um auf die PLC-Variablentabelle zuzugreifen, ändern Sie folgenden Programmcode:

```
//Gives individual access to a specific Plc tag table
private static void AccessToPlcTagTableWithFind(PlcSoftware plcSoftware)
{
    PlcTagTableComposition tagTables = plcSoftware.TagTableGroup.TagTables;
    // alternatively, PlcTagTableComposition tagTables =
plcSoftware.TagTableGroup.Groups.Find("UserGroup XYZ").TagTables;
    PlcTagTable controllerTagTable = tagTables.Find("Tag table XYZ");
    // The parameter specifies the name of the tag table
}
```

7.18.14.8 Informationen einer PLC-Variablentabelle abfragen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Über PLC-Variablentabellen können Sie auf Anwenderkonstanten, Systemkonstanten und Variablen zugreifen. Die Anzahl der Variablenzusammensetzung einer Variablentabelle entspricht der Anzahl der Variablen in der Variablentabelle. Die PLCTagTable enthält die folgenden Navigatoren, Attribute und Aktionen.

In der PLC-Variablentabelle wird auf die folgenden Attribute zugegriffen.

Name	Typ	Typ
IsDefault	Bool	Schreibgeschützt
ModifiedTimeStamp	DateTime	Schreibgeschützt
Name	String	Schreibgeschützt

Die PLC-Variablentabelle enthält die nachstehend aufgeführten Aktionen.

Name	Rückgabetyp	Beschreibung
Delete	leer	Löscht die Instanz. Löst eine Ausnahme aus, wenn IsDefault wahr ist.
Exportieren	leer	Exportiert SIMATIC ML einer PLC-Variablentabelle.
ShowInEditor	leer	Zeigt die Variablentabelle im PLC-Variablentabelleneditor an.

Programmcode

Um die Informationen einer PLC-Variablentabelle abzufragen, ändern Sie den folgenden Programmcode:

```
private static void AccessPlcConstantsUsingFind(PlcTagTable tagTable)
{
    PlcUserConstantComposition plcUserConstants = tagTable.UserConstants;
    PlcUserConstant plcUserConstant = plcUserConstants.Find("Constant XYZ");
    //PlcSystemConstantComposition plcSystemConstants = tagTable.SystemConstants;
    //PlcSystemConstant plcSystemConstant = plcSystemConstants.Find("Constant XYZ");
}
private static void EnumeratePlcTags(PlcTagTable tagTable)
{
    PlcTagComposition plcTags = tagTable.Tags;
    foreach (PlcTag plcTag in plcTags)
    {
        string name = plcTag.Name; string typeName = plcTag.DataTypeName;
        string logicalAddress = plcTag.LogicalAddress;
    }
}
private static void EnumeratePlcTagsUsingFind(PlcTagTable tagTable)
{
    PlcTagComposition plcTags = tagTable.Tags;
    PlcTag plcTag = plcTags.Find("Constant XYZ");
}
```

7.18.14.9 Zeitpunkt der letzten Änderung einer PLC-Variablentabelle lesen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Das Format des Zeitstempels ist UTC.

Programmcode

Um den Zeitstempel einer bestimmten PLC-Variablentabelle zu lesen, ändern Sie den folgenden Programmcode:

```
//Reads Time-Stamp of a plc Tag Table
private static void GetLastModificationDateOfTagtable(PlcSoftware plcSoftware)
{
    PlcTagTable plcTagTable = plcSoftware.TagTableGroup.TagTables.Find("MyTagTable");
    DateTime modifiedTagTableTimeStamp = plcTagTable.ModifiedTimeStamp;
}
```

7.18.14.10 PLC-Variablentabelle aus einer Gruppe löschen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode

Ändern Sie den folgenden Programmcode, um eine bestimmte Variablentabelle aus einer Gruppe zu löschen:

```
//Deletes a PlcTagTable of a group
private static void DeletePlcTagTableInAGroup(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup group = plcSoftware.TagTableGroup;
    PlcTagTable tagtable = group.TagTables.Find("MyTagTable");
    if (tagtable!= null)
    {
        tagtable.Delete();
    }
}
```

7.18.14.11 PLC-Variablen enumerieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Programmcode: PLC-Variablen in Variablentabellen enumerieren

Um alle PLC-Variablen in einer Variablentabelle zu enumerieren, ändern Sie den folgenden Programmcode:

```
//Enumerates all plc tags in a specific tag table
private static void EnumerateAllPlcTagsInTagTable(PlcSoftware plcSoftware)
{
    PlcTagTable tagTable = plcSoftware.TagTableGroup.TagTables.Find("Tagtable XYZ");
    foreach (PlcTag tag in tagTable.Tags)
    {
        // add code here
    }
}
```

7.18.14.12 Auf PLC-Variablen zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Der Typ `PlcTagComposition` stellt eine Sammlung von PLC-Variablen dar.

Programmcode: Auf eine bestimmte PLC-Variable zugreifen

Um auf die gewünschte PLC-Variable zuzugreifen, ändern Sie folgenden Programmcode. Sie haben Zugriff auf die folgenden Attribute:

- Name (nur Lesezugriff)
- Name des Datentyps
- Logische Adresse
- Kommentar
- 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
}
```

Programmcode: Variablen anlegen

Ändern Sie folgenden Programmcode:

```
private static void CreateTagInPLCTagtable(PlcSoftware plcsoftware)
// Create a tag in a tag table with default attributes
{
    string tagName = "MyTag";
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcTagComposition tagComposition = table.Tags;
    PlcTag tag = tagComposition.Create(tagName);
}
```

Ändern Sie folgenden Programmcode:

```
private static void CreateTagInPLCTagtable(PlcSoftware plcsoftware)
// Create a tag of data type bool and logical address not set
{
    string tagName = "MyTag";
    string dataType = "Bool";
    string logicalAddress ="";
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcTagComposition tagComposition = table.Tags;
    PlcTag tag = tagComposition.Create(tagName, dataType, logicalAddress);
}
```

Programmcode: Variablen löschen

Ändern Sie folgenden Programmcode:

```
private static void DeleteTagFromPLCTagtable(PlcSoftware plcsoftware)
// Deletes a single tag of a tag table
{
    string tagName = "MyTag";
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcTagComposition tagComposition = table.Tags;
    PlcTag tag = tagComposition.Find(tagName);
    if (tag != null)
    {
        tag.Delete();
    }
}
```

7.18.14.13 Auf PLC-Konstanten zugreifen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Anwendung

Der Typ `PlcUserConstantComposition` stellt eine Sammlung von PLC-Benutzerkonstanten dar. Sie haben Zugriff auf die folgenden Attribute:

- Name (nur Lesezugriff)
- Name des Datentyps
- Wert

Der Typ `PlcSystemConstantComposition` stellt eine Sammlung von PLC-Systemkonstanten dar. Sie haben Zugriff auf die folgenden Attribute:

- Name (nur Lesezugriff)
- Datentypname (nur Lesezugriff)
- Wert (nur Lesezugriff)

Programmcode: Benutzerkonstanten anlegen

Ändern Sie folgenden Programmcode:

```
private static void CreateUserConstantInPLCTagtable(PlcSoftware plcsoftware)
// Create a user consrant in a tag table
{
    string constantName = "MyConstant";
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcUserConstantComposition userConstantComposition = table.UserConstants;
    PlcUserComnstant userConstant = userConstantComposition.Create(constantName);
}
```

Programmcode: Benutzerkonstanten löschen

Ändern Sie folgenden Programmcode:

```
private static void DeleteUserConstantFromPLCTagtable(PlcSoftware plcsoftware)
// Deletes a single user constant of a tag table
{
    PlcTagTable table = plcsoftware.TagTableGroup.TagTables.Find("myTagTable");
    PlcUserConstantComposition userConstantComposition = table.UserConstants;
    PlcUserConstant userConstant = userConstantComposition.Find("MyConstant");
    if (userConstant != null)
    {
        userConstant.Delete();
    }
}
```

Programmcode: Auf Systemkonstanten zugreifen

Ändern Sie folgenden Programmcode:

```
//Gives individual access to a specific system constant
private static void AccessSystemConstant(PlcTagTable tagTable)
{
    PlcTag systemConstant = tagTable.SystemConstants.Find("Constant XYZ");
    // The parameter specifies the name of the tag
}
```

Siehe auch

Benutzerdefinierte Gruppen für PLC-Variablen erzeugen (Seite 369)
Benutzerdefinierte Gruppen für PLC-Variablen löschen (Seite 370)
PLC-Variablentabelle aus einer Gruppe löschen (Seite 373)
Auf PLC-Variablen zugreifen (Seite 375)
Starten des PLC-Variableneditors (Seite 366)
Zeitpunkt der letzten Änderung einer PLC-Variablentabelle lesen (Seite 373)

7.19 Funktionen auf OPC

7.19.1 Konfiguration des sicheren Kommunikationsprotokolls für den OPC UA-Server

Voraussetzungen

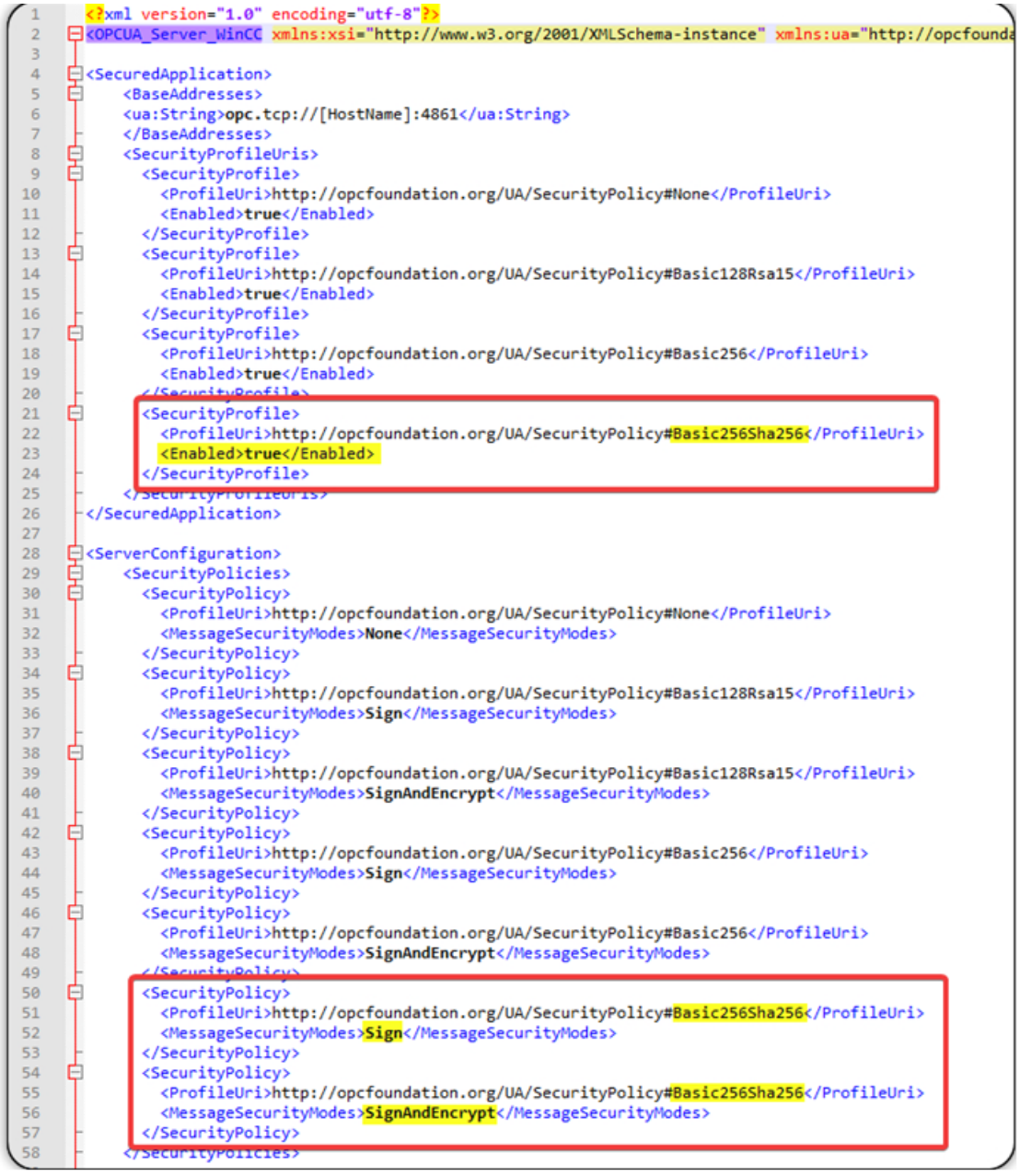
- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Einleitung

Sie können mit der TIA Portal Openness-Anwendung den OPC UA-Server mit der Sicherheitsrichtlinie Basic256Sha256 konfigurieren. Die Sicherheitsrichtlinie Basic256Sha256 muss in den Runtime-Einstellungen hinzugefügt werden. RDP muss die Eigenschaften in der xml-Konfigurationsdatei übersetzen.

Die Standardwerte lauten Enabled, Sign sowie Sign and Encrypt.

In der XML-Datei <Project>\OPC\uaserver\OPCUaServerWinCCPro.xml müssen Sie die Sicherheitsrichtlinie sowie Sicherheitsrichtlinien gemäß der ES-Gerätekonfiguration festlegen.



```
<?xml version="1.0" encoding="utf-8"?>
<OPCUA_Server_WinCC xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ua="http://opcfounda

<SecuredApplication>
    <BaseAddresses>
    <ua:String>opc.tcp://[HostName]:4861</ua:String>
    </BaseAddresses>
    <SecurityProfileUris>
      <SecurityProfile>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#None</ProfileUri>
        <Enabled>true</Enabled>
      </SecurityProfile>
      <SecurityProfile>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic128Rsa15</ProfileUri>
        <Enabled>true</Enabled>
      </SecurityProfile>
      <SecurityProfile>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256</ProfileUri>
        <Enabled>true</Enabled>
      </SecurityProfile>
      <SecurityProfile>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256Sha256</ProfileUri>
        <Enabled>true</Enabled>
      </SecurityProfile>
    </SecurityProfileUris>
</SecuredApplication>

<ServerConfiguration>
    <SecurityPolicies>
      <SecurityPolicy>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#None</ProfileUri>
        <MessageSecurityModes>None</MessageSecurityModes>
      </SecurityPolicy>
      <SecurityPolicy>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic128Rsa15</ProfileUri>
        <MessageSecurityModes>Sign</MessageSecurityModes>
      </SecurityPolicy>
      <SecurityPolicy>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic128Rsa15</ProfileUri>
        <MessageSecurityModes>SignAndEncrypt</MessageSecurityModes>
      </SecurityPolicy>
      <SecurityPolicy>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256</ProfileUri>
        <MessageSecurityModes>Sign</MessageSecurityModes>
      </SecurityPolicy>
      <SecurityPolicy>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256</ProfileUri>
        <MessageSecurityModes>SignAndEncrypt</MessageSecurityModes>
      </SecurityPolicy>
      <SecurityPolicy>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256Sha256</ProfileUri>
        <MessageSecurityModes>Sign</MessageSecurityModes>
      </SecurityPolicy>
      <SecurityPolicy>
        <ProfileUri>http://opcfoundation.org/UA/SecurityPolicy#Basic256Sha256</ProfileUri>
        <MessageSecurityModes>SignAndEncrypt</MessageSecurityModes>
      </SecurityPolicy>
    </SecurityPolicies>
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

7.19.2 Sicherheitsrichtlinie für OPC UA festlegen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Öffnen eines Projekts (Seite 99)
- OPC UA Server ist aktiviert

Anwendung

Sie können mit der TIA Portal Openness-Anwendung die Sicherheitsrichtlinie in OPC UA festlegen. Sie können die Sicherheitsrichtlinie als dynamisches Attribut vom Typ markierte Enumeration implementieren: OpcUaSecurityPolicies. Die Sicherheitsrichtlinie ist in TIA Portal Openness nur verfügbar, wenn der OPC UA Server aktiviert ist. Wenn der OPC UA Server deaktiviert ist, wird bei dem Versuch, wegen eines anderen, nicht verfügbaren Attributs auf die Sicherheitsrichtlinie zuzugreifen, eine EngineeringNotSupportedException ausgelöst.

Die nachstehende Tabelle zeigt die möglichen Werte, die für Sicherheitsrichtlinien zu finden sind:

Name im TIA UI	Enumerations-Eintrag	Wert	Anmerkungen
-	NoneSelected	0	Entspricht der TIA UI, wenn kein Kontrollkästchen markiert ist.
No security	OpcUaSecurityPolicies-None	1	
Basic128Rsa15 - Sign	OpcUaSecurityPolicies128RSAS	2	
Basic128Rsa15 - Sign & Encrypt	OpcUaSecurityPolicies128RSASE	4	
Basic256 - Sign	OpcUaSecurityPolicies256S	8	
Basic256 - Sign & Encrypt	OpcUaSecurityPolicies256SE	16	
Basic256Sha256 - Sign	OpcUaSecurityPolicies256SHAS	32	
Basic256Sha256 - Sign & Encrypt	OpcUaSecurityPolicies256SHASE	64	

Programmcode

Um die Sicherheitsrichtlinie in OPC UA mit TIA Portal Openness festzulegen, ändern Sie folgenden Programmcode:

```
DeviceItem UpcUaSubmodule= ...;  "
object SecurityPolicies = UpcUaSubmodule.GetAttribute("OpcUaSecurityPolicies");
if(SecurityPolicies | OpcUaSecurityPolicies.OpcUaSecurityPolicies256S ==
OpcUaSecurityPolicies.OpcUaSecurityPolicies256S)
{
//Do something
}
UpcUaSubmodule.SetAttribute("OpcUaSecurityPolicies",
OpcUaSecurityPolicies.OpcUaSecurityPolicies256S |
OpcUaSecurityPolicies.OpcUaSecurityPolicies256SHASE);
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

7.20 SiVArc Openness

7.20.1 Einleitung

7.21 Openness für CP 1604/CP 1616/CP 1626

Allgemein

Mithilfe der TIA Portal Openness-Anwendung können Sie die Transferbereiche und die Zuordnungsregeln der Transferbereiche konfigurieren. Dies gilt für die Kommunikationsprozessoren CP 1604/CP 1616 ab V2.8 (und je nach Artikelnummer auch ab V2.7) sowie CP 1626 ab V1.1.

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe "Verbindung zum TIA Portal herstellen".
- Ein Projekt ist geöffnet.
  Siehe "Projekt öffnen".
- Vor dem Übersetzen des Projekts müssen alle Geräte "offline" sein.

Konfiguration von Transferbereichen

Transferbereiche erstellen

Um z. B. einen Transferbereich vom Typ "CD" für einen CP 1604 zu erstellen, verwenden Sie folgenden Programmcode:

```
NetworkInterface cpItf = CP
1604Interface.GetService<NetworkInterface>();

// Transferbereiche erstellen

TransferAreaComposition transferAreas = cpItf.TransferAreas;


// Einfacher Transferbereich vom Typ Eingang

TransferArea transferAreaInput =
  transferAreas.Create("Input CD", TransferAreaType.CD);
```

Attribut	Beschreibung	
name	Gibt den Namen des zu erstellenden Transferbereichs an.	
type	Gibt den Typ des zu erstellenden Transferbereichs an. Folgende Typen sind möglich:	
	TransferAreaType.CD	Steuerungsgerät für Datenaustausch
	TransferAreaType.F_PS	Datenaustausch PROFIsafe
	TransferAreaType.TM	Zuordnung des Transfermoduls
	Hinweis: Es ist nicht möglich, den Typ später zu ändern.	

Attribute der Transferbereiche festlegen

Um die Attribute eines Transferbereichs festzulegen, ändern Sie z. B. folgenden Programmcode:

```
transferAreaTm.LocalToPartnerLength = 8;
transferAreaTm.Direction = TransferAreaDirection.LocalToPartner;
string name = transferAreaTm.Name
```

Einige Attribute müssen festgelegt oder abgefragt werden, alle Attribute können jedoch mit den allgemeinen Aufrufen "GetAttribute()" oder "SetAttribute()" festgelegt oder abgefragt werden. Verwenden Sie z. B. folgenden Programmcode:

```
const string myIndividualComment = "MyIndividualComment";
transferAreaTm.SetAttribute("Comment", myIndividualComment);
Int32 updateTime = transferAreaTm.GetAttribute("TransferUpdateTime")
```

Attribut	Beschreibung	
Name (Zeichenkette)	Gibt den Namen des Transferbereichs an.	
Richtung	Gibt die Richtung an, in der die Daten des Transferbereichs übertragen werden. Folgende Richtungen sind möglich:	
	TransferAreaDirection.LocalTo-Partner	Daten des Transferbereichs werden vom IO-Device an die überlagerte Steuerung übertragen.
	TransferAreaDirection.partnerTo-Local	Daten des Transferbereichs werden von der überlagerten Steuerung an das IO-Device übertragen.
	TransferAreaDirection.bidirectional	Daten des Transferbereichs können zwischen der überlagerten Steuerung und dem IO-Device in beiden Richtungen übertragen werden. Mit dem Attribut "LocalToPartnerLength" wird die Länge angegeben, mit der Daten des Transferbereichs vom IO-Device an die überlagerte Steuerung übertragen werden. Das Attribut "PartnerToLocalLength" legt die Datenmenge fest, die von der überlagerten Steuerung an das IO-Device übertragen werden kann.
Comment (Zeichenkette)	Textfeld für Kommentar zum Transferbereich.	
LocalToPartnerLength	Gibt die Länge der Daten des Transferbereichs an, die vom IO-Device an die überlagerte Steuerung übertragen werden.	
PartnerToLocalLength	Gibt die Länge der Daten des Transferbereichs an, die von der überlagerten Steuerung an das IO-Device übertragen werden.	
LocalAdresses	Gibt die Eingangs- und Ausgangsadressen des Transferbereichs im lokalen Gerät an.	
PartnerAdresses	Gibt die Eingangs- und Ausgangsadressen des Transferbereichs in der überlagerten Steuerung an.	
TransferUpdateTime(Int32)	Gibt die Aktualisierungszeit des Transferbereichs an. Nur bei einem Transferbereich vom Typ "TransferAreaType.TM“ festgelegt oder abgefragt.	
PositionNumber	Gibt die Anzahl der virtuellen Submodule dieses Transferbereichs an.	
Typ	Gibt den Typ des Transferbereichs an, schreibgeschützt.	
TransferAreaMappingRules	Gibt die Routing-Tabelle des Routing-Bereichs an, schreibgeschützt.	

Transferbereiche löschen

Verwenden Sie zum Löschen von Transferbereichen folgenden Programmcode:

```
transferAreaInput.Delete();
```

Iteration über Transferbereiche

Verwenden Sie zum Iterieren von Transferbereichen folgenden Programmcode:

```
TransferAreaComposition transferAreas = cpItf.TransferAreas;
  foreach (TransferArea transferArea in transferAreas)
  {
         transferArea.Delete();
 }
```

Konfiguration des E/A-Routing

E/A-Routen erstellen

Verwenden Sie zum Erstellen von E/A-Routen folgenden Programmcode:

```
// TransferAreaMappingRule erstellen
TransferAreaMappingRuleComposition routingTable
                          = transferArea.TransferAreaMappingRules;


// Neue E/A-Route erstellen
TransferAreaMappingRule route1 =
    routingTable.Create();
```

Attribute von E/A-Routen festlegen

Die folgenden Attribute können für das E/A-Routing festgelegt werden:

Attribut	Beschreibung
Offset	In Bit angegebener Offset im Routing-Bereich, dem die Daten zugewiesen werden sollen. Die Länge des Offsets laut Festlegung durch die Attribute "Begin" und "End".
Ziel	Gibt das Modul oder Submodul des IO-Device an, das die Daten enthält, die der Konfiguration des IO-Device zugeordnet werden sollen, d. h. einen Transferbereich vom Typ "TM".
IoType	Das Attribut "IoType" kann nur in einem Transferbereich vom Typ "Input" geändert werden. Zusätzlich muss ein gemischtes Modul für diesen Transferbereich als "Ziel" konfiguriert werden. Erst kann können Sie auswählen, ob die Daten der Eingänge (IoType.Input) zu lesen sind oder ob die Daten der Ausgänge (IoType.Output) (zurück-)gelesen werden sollen.
Begin	Gibt den Anfang der mit dem Attribut "Target" zu lesenden Daten an.
End	Gibt das Ende der mit dem Attribut "Target" zu lesenden Daten an.

E/A-Routen löschen

Verwenden Sie zum Löschen von E/A-Routen folgenden Programmcode:

```
transferAreaMappingRule.Delete();
```

Iteration über E/A-Routing

Für eine Iteration über E/A-Routing verwenden Sie folgenden Programmcode:

```
TransferAreaMappingRuleComposition routingTable =
                transferArea.TransferAreaMappingRules;
  foreach (TransferAreaMappingRule route in routingTable)
  {
         route.Delete();
  }
```

7.22 Openness für SIMATIC Ident

7.22.1 Openness für SIMATIC Ident

Die SIMATIC Ident-Geräte (Kommunikationsmodule, RF600-Reader und MV400/MV500-Lesegeräte) können mit Hilfe von TIA Portal Openness konfiguriert werden. Die nachfolgenden Kapitel geben Ihnen einen Überblick über die baugruppen-spezifischen Parameter.

Erforderliche Grundkenntnisse

Zum Verständnis der nachfolgenden Inhalte sind allgemeine Kenntnisse auf dem Gebiet der Automatisierungstechnik und Identifikationssysteme sowie mit TIA Portal Openness erforderlich.

Weiterführende Informationen

Ausführliche Informationen zu TIA Portal Openness finden Sie in der Hilfe "Projekte über Skripte automatisieren". Diese Hilfe enthält auch konkrete Programmierbeispiele (vergleiche Kapitel "Funktionen der Projekte und Projektdaten" und "Funktionen für das Aufrufen von Geräten, Netzwerken und Verbindungen").

Voraussetzung

- Die Openness-Anwendung ist mit dem TIA Portal verbunden.
- Ein Projekt ist geöffnet.
- Für das Übersetzen des Projekts müssen alle Geräte offline sein.

7.22.2 ASM 456

Nachfolgend werden die Openness-Parameter für die Baugruppe "ASM 456" beschrieben.

Tabelle 7-1 Parameter der Baugruppe "ASM 456"; Modul "Word: 2 IN/OUT DP-V1"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
User Mode	UserMode	ulong	Ident-Profil/ RFID-Normprofil	3	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter wählen Sie den Baustein aus: • Ident-Profil/RFID-Normprofil: In der Steuerung kommt der Programmbaustein für das Ident-Profil/RFID-Normprofil zum Einsatz. • FB 45 / FC 45: Singletag-Betrieb. In der Steuerung wird der FB 45 (PROFIBUS/PROFINET) oder der FC 45 (PROFIBUS) eingesetzt. • FB 55 / FC 55: Multitag-Betrieb. In der Steuerung wird der FB 55 (PROFIBUS/PROFINET) oder der FC 55 (PROFIBUS) eingesetzt. • FC 56 Filehandler für S7-300 und S7-400
			FB 45 / FC 45	1	R/W	
			FB 55 / FC 55	4	R/W	
			FC 56	5	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
MOBY Mode	MobyMode	ulong	RF200/ RF300/ RF600; MV4x0; MOBY U/D	5	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Betriebsart des Kommunikationsmoduls ein. • RF200/RF300/RF600; MV4x0; MOBY U/D • MOBY I/E Normaladressierung • RF300 Filehandler • MOBY U Filehandler • MOBY I Filehandler Normaladressierung: Der Transponder wird mit physikalischen Adressen angesprochen. Filehandler: Der Transponder muss vor der Benutzung formatiert werden.
			MOBY I/E Normaladressierung	1	R/W	
			RF300 Filehandler	149	R/W	
			MOBY U Filehandler	133	R/W	
			MOBY I Filehandler	129	R/W	
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Diagnosemeldungen	Diagnostic-Messages	int	Keine	1	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie ein, ob Hardware-Diagnosemeldungen gemeldet werden sollen. • Keine: Neben der Standarddiagnose werden keine weiteren Alarme generiert. • Hard Errors: Schwerwiegende Hardware-Fehler werden über die S7-Diagnose gemeldet. • Hard/Soft Errors niederprior Schwerwiegende Hardware-Fehler und Fehler die während der Befehlsbearbeitung auftreten, werden über die S7-Diagnose gemeldet. Das "Ext_Diag"-Bit wird nicht gesetzt. • Hard/Soft Errors hochprior Schwerwiegende Hardware-Fehler und Fehler die während der Befehlsbearbeitung auftreten, werden über die S7-Diagnose gemeldet. Das "Ext_Diag"-Bit wird gesetzt.
			Hard Errors	2	R/W	
			Hard/Soft Errors niederprior	3	R/W	
			Hard/Soft Errors hochprior	5	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Unterdrückung Error-LED	SuppressionOfErrorLed	int	Keine	0	R/W	Deaktivieren der Error-LED (ERR) eines Kanals. Das Kommunikationsmodul verfügt über zwei Kanäle, an die Reader / optischen Lesegeräte angeschlossen werden können. Wird nur einer der Kanäle verwendet, blinkt die Error-LED des anderen Kanals permanent. Mithilfe der Unterdrückung können Sie die Error-LED des nicht verwendeten Kanals deaktivieren.
			Kanal 1	1	R/W	
			Kanal 2	2	R/W	

Tabelle 7-2 Parameter der Baugruppe "ASM 456"; Modul "RF680R/RF685R"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
User Mode	UserMode	ulong	Ident-Profil/ RFID-Normprofil	3	R	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter wählen Sie den Baustein aus: • Ident-Profil/RFID-Normprofil: In der Steuerung kommt der Programmbaustein für das Ident-Profil/RFID-Normprofil zum Einsatz.
MOBY Mode	MobyMode	ulong	RF680R/ RF685R	32	R	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Betriebsart des Kommunikationsmoduls ein. • RF680R/RF685R Normaladressierung: Der Transponder wird mit physikalischen Adressen angesprochen. Filehandler: Der Transponder muss vor der Benutzung formatiert werden.

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Übertragungsgeschwindigkeit	BaudRate	ulong	115,2 kBd	14	R	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Diagnosemeldungen	Diagnostic-Messages	int	Keine	1	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie ein, ob Hardware-Diagnosemeldungen gemeldet werden sollen. • Keine: Neben der Standarddiagnose werden keine weiteren Alarme generiert. • Hard Errors: Schwerwiegende Hardware-Fehler werden über die S7-Diagnose gemeldet. • Hard/Soft Errors niederprior Schwerwiegende Hardware-Fehler und Fehler die während der Befehlsbearbeitung auftreten, werden über die S7-Diagnose gemeldet. Das "Ext_Diag"-Bit wird nicht gesetzt. • Hard/Soft Errors hochprior Schwerwiegende Hardware-Fehler und Fehler die während der Befehlsbearbeitung auftreten, werden über die S7-Diagnose gemeldet. Das "Ext_Diag"-Bit wird gesetzt.
			Hard Errors	2	R/W	
			Hard/Soft Errors niederprior	3	R/W	
			Hard/Soft Errors hochprior	5	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Unterdrückung Error-LED	SuppressionOfErrorLed	int	Keine	0	R/W	Deaktivieren der Error-LED (ERR) eines Kanals. Das Kommunikationsmodul verfügt über zwei Kanäle, an die Reader / optischen Lesegeräte angeschlossen werden können. Wird nur einer der Kanäle verwendet, blinkt die Error-LED des anderen Kanals permanent. Mithilfe der Unterdrückung können Sie die Error-LED des nicht verwendeten Kanals deaktivieren.
			Kanal 1	1	R/W	
			Kanal 2	2	R/W	

7.22.3 ASM 475

Nachfolgend werden die Openness-Parameter für die Baugruppe "ASM 475" beschrieben.

Tabelle 7-3 Parameter der Baugruppe "ASM 475"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
MOBY Mode	MobyMode	ulong	MOBY I/E/F Normaladressierung	1	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Betriebsart des Kommunikationsmoduls ein. • MOBY I/E/F Normaladressierung • MOBY I Filehandler • RF200/RF300/RF600; MOBY U/D Normaladressierung • MOBY U Filehandler Normaladressierung: Der Transponder wird mit physikalischen Adressen angesprochen. Filehandler: Der Transponder muss vor der Benutzung formatiert werden.
			MOBY I Filehandler	129	R/W	
			RF200/ RF300/ RF600; MOBY U/D Normaladressierung	5	R/W	
			MOBY U Filehandler	133	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Übertragungsgeschwindigkeit	BaudRate	ulong	19200 Bd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57600 Bd	13	R/W	
			115200 Bd	14	R/W	

7.22.4 RF120C

Nachfolgend werden die Openness-Parameter für die Baugruppe "RF120C" beschrieben.

Tabelle 7-4 Parameter der Baugruppe "RF120C"; Parameter der Parametergruppe "Reader"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Diagnosemeldungen	Diagnostic-Messages	int	Keine	1	R/W	Mit diesem Parameter stellen Sie ein, ob Hardware-Diagnosemeldungen gemeldet werden sollen. • Keine: Neben der Standarddiagnose werden keine weiteren Alarme generiert. • Hard Errors: Schwerwiegende Hardware-Fehler werden über die S7-Diagnose gemeldet.
			Hard Errors	2	R/W	
User Mode	UserMode	ulong	Ident-Profil	3	R	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter wählen Sie den Baustein aus: • Ident-Profil: In der Steuerung kommt der Programmbaustein für das Ident-Profil zum Einsatz.

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Ident-Gerät/-System	IdentDeviceOrSystem	int	RF200 allgemein	53	R/W	Auswahl des angeschlossenen Ident-Geräts/-Systems. Abhängig von der getroffenen Auswahl, wird die Parametergruppe "Ident-System" entsprechend angepasst.
			RF290R	69	R/W	
			RF300 allgemein	85	R/W	
			RF380R	101	R/W	
			MOBY U	5	R/W	
			Allgem. Reader	197	R/W	
			Parameter über FB / Optische Lesegeräte	214	R/W	
			RF600 [1]	117	R/W	
			SLG D10S [1]	21	R/W	
			SLG D11S/ D12S [1]	37	R/W	

[1] Die Sub-Parameter dieser Parameter werden durch Openness nicht unterstützt.

Nachfolgend werden die Openness-Parameter der Parametergruppe "Ident-Gerät/-System" beschrieben.

Tabelle 7-5 Parameter der Baugruppe "RF120C"; Parameter der Parametergruppe "Ident-Gerät/-System: RF200 allgemein"

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Beachten Sie, dass der hier angegebene Wert automatisch aus der Gerätekonfiguration der angeschlossenen Geräte übernommen wird. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	
Anwesenheitskontrolle	Presence-Check	ulong	An	1	R/W	An = Sobald sich ein Transponder im Antennenfeld des Readers befindet, wird dessen Anwesenheit gemeldet. Aus = Die Anwesenheitsanzeige am FB wird unterdrückt. Die Antenne am Reader ist jedoch eingeschaltet, solange diese nicht durch einen Befehl abgeschaltet wurde.
			Aus	0	R/W	
ERR-LED zurücksetzen	ResetError-LED	int	An	2	R/W	An = Das Blinken der Error-LED am Kommunikationsmodul wird durch jeden Reset des FBs zurückgesetzt. Aus = An der Error-LED wird immer der letzte Fehler angezeigt. Ein Rücksetzen der Anzeige ist nur durch Ausschalten des Kommunikationsmoduls möglich.
			Aus	0	R/W	

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Max. Transponder-Anzahl	MaxNoOfTransponders	int	1	<number>	R	Anzahl der erwarteten Transponder im Antennenfeld. Die Auswahl ist abhängig vom angeschlossenen Gerät.
Transponder-Typ	TransponderType	int	ISO 15693	1	R	Auswahl der verwendeten Transponder-Typen. Die Auswahl ist abhängig vom angeschlossenen Gerät.

Tabelle 7-6 Parameter der Baugruppe "RF120C"; Parameter der Parametergruppe "Ident-Gerät/-System: RF290R"

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Beachten Sie, dass der hier angegebene Wert automatisch aus der Gerätekonfiguration der angeschlossenen Geräte übernommen wird. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	
Anwesenheitskontrolle	Presence-Check	ulong	An	1	R/W	An = Sobald sich ein Transponder im Antennenfeld des Readers befindet, wird dessen Anwesenheit gemeldet. Aus = Die Anwesenheitsanzeige am FB wird unterdrückt. Die Antenne am Reader ist jedoch eingeschaltet, solange diese nicht durch einen Befehl abgeschaltet wurde.
			Aus	0	R/W	

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
ERR-LED zurücksetzen	ResetError-LED	int	An	2	R/W	An = Das Blinken der Error-LED am Kommunikationsmodul wird durch jeden Reset des FBs zurückgesetzt. Aus = An der Error-LED wird immer der letzte Fehler angezeigt. Ein Rücksetzen der Anzeige ist nur durch Ausschalten des Kommunikationsmoduls möglich.
			Aus	0	R/W	
HF-Leistung	RfPower	double	0,50 ... 5,00	<text>	R/W	Einstellung für die Ausgangsleistung des Readers. Die auswählbaren Werte sind abhängig vom angeschlossenen Gerät.
Max. Transponder-Anzahl	MaxNoOfTransponders	int	1	<number>	R	Anzahl der erwarteten Transponder im Antennenfeld. Die Auswahl ist abhängig vom angeschlossenen Gerät.
Transponder-Typ	TransponderType	int	ISO 15693	1	R	Auswahl der verwendeten Transponder-Typen. Die Auswahl ist abhängig vom angeschlossenen Gerät.

Tabelle 7-7 Parameter der Baugruppe "RF120C"; Parameter der Parametergruppe "Ident-Gerät/-System: RF300 allgemein"

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Beachten Sie, dass der hier angegebene Wert automatisch aus der Gerätekonfiguration der angeschlossenen Geräte übernommen wird. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	
Anwesenheitskontrolle	Presence-Check	ulong	An	1	R/W	An = Sobald sich ein Transponder im Antennenfeld des Readers befindet, wird dessen Anwesenheit gemeldet. Aus (HF-Feld an) = Die Anwesenheitsanzeige am FB wird unterdrückt. Die Antenne am Reader ist jedoch eingeschaltet, solange diese nicht durch einen Befehl abgeschaltet wurde. Aus (HF-Feld aus) = Die Antenne wird nur eingeschaltet, wenn ein Befehl abgeschickt wird und schaltet sich danach wieder ab.
			Aus (HF-Feld an)	3	R/W	
			Aus (HF-Feld aus)	2	R/W	

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
ERR-LED zurücksetzen	ResetError-LED	int	An	2	R/W	An = Das Blinken der Error-LED am Kommunikationsmodul wird durch jeden Reset des FBs zurückgesetzt. Aus = An der Error-LED wird immer der letzte Fehler angezeigt. Ein Rücksetzen der Anzeige ist nur durch Ausschalten des Kommunikationsmoduls möglich.
			Aus	0	R/W	
Max. Transponder-Anzahl	MaxNoOfTransponders	int	1	<number>	R	Anzahl der erwarteten Transponder im Antennenfeld. Die Auswahl ist abhängig vom angeschlossenen Gerät.
Transponder-Typ	TransponderType	int	RF300	5	R/W	Auswahl der verwendeten Transponder-Typen. Die Auswahl ist abhängig vom angeschlossenen Gerät.
			ISO 15693	1	R/W	

Tabelle 7-8 Parameter der Baugruppe "RF120C"; Parameter der Parametergruppe "Ident-Gerät/-System: RF380R"

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Beachten Sie, dass der hier angegebene Wert automatisch aus der Gerätekonfiguration der angeschlossenen Geräte übernommen wird. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Anwesenheitskontrolle	PresenceCheck	ulong	An	1	R/W	An = Sobald sich ein Transponder im Antennenfeld des Readers befindet, wird dessen Anwesenheit gemeldet. Aus (HF-Feld an) = Die Anwesenheitsanzeige am FB wird unterdrückt. Die Antenne am Reader ist jedoch eingeschaltet, solange diese nicht durch einen Befehl abgeschaltet wurde. Aus (HF-Feld aus) = Die Antenne wird nur eingeschaltet, wenn ein Befehl abgeschickt wird und schaltet sich danach wieder ab.
			Aus (HF-Feld an)	3	R/W	
			Aus (HF-Feld aus)	2	R/W	
ERR-LED zurücksetzen	ResetErrorLED	int	An	2	R/W	An = Das Blinken der Error-LED am Kommunikationsmodul wird durch jeden Reset des FBs zurückgesetzt. Aus = An der Error-LED wird immer der letzte Fehler angezeigt. Ein Rücksetzen der Anzeige ist nur durch Ausschalten des Kommunikationsmoduls möglich.
			Aus	0	R/W	
HF-Leistung	RfPower	double	0,50 ... 5,00	<text>	R/W	Einstellung für die Ausgangsleistung des Readers. Die auswählbaren Werte sind abhängig vom angeschlossenen Gerät.
Max. Transponder-Anzahl	MaxNoOfTransponders	int	1	<number>	R	Anzahl der erwarteten Transponder im Antennenfeld. Die Auswahl ist abhängig vom angeschlossenen Gerät.
Transponder-Typ	TransponderType	int	RF300	5	R/W	Auswahl der verwendeten Transponder-Typen. Die Auswahl ist abhängig vom angeschlossenen Gerät.
			ISO 15693	1	R/W	

Tabelle 7-9 Parameter der Baugruppe "RF120C"; Parameter der Parametergruppe "Ident-Gerät/-System: Allgem. Reader"

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Beachten Sie, dass der hier angegebene Wert automatisch aus der Gerätekonfiguration der angeschlossenen Geräte übernommen wird. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	
Byte-Sequenz des Reset-Parameters	ByteSequenceHex	string	[1]	<text>	R/W	Byte-Sequenz des Reset-Parameters des Readers.

[1] Eine ausführliche Beschreibung zu diesem Parameter finden Sie in dem nachfolgenden Absatz.

Tabelle 7-10 Parameter der Baugruppe "RF120C"; Parameter der Parametergruppe "Ident-Gerät/-System: Parameter über FB / Optische Lesesysteme"

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Beachten Sie, dass der hier angegebene Wert automatisch aus der Gerätekonfiguration der angeschlossenen Geräte übernommen wird. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	
MOBY Mode	MobyMode	ulong	RF200/ RF300/ RF600; MV4x0; MOBY U/D	5	R	Mit diesem Parameter stellen Sie die Betriebsart des Kommunikationsmoduls ein.

Der Parameter "Byte-Sequenz des Reset-Parameters"

Über diese Funktion können Sie die Reset-Parameter mit Hilfe eines Byte-Arrays (16) angeben. Diese Einstellung richten sich ausschließlich an geschulte Anwender.

Folgende Reset-Parameter stehen zur Verfügung:

Tabelle 7-11 Reset-Parameter

Byte	1	2...5	6	7...8	9	10	11	12	13...14	15	16
Wert	4	0	10	0	scanning_ time	param	option_1	distance_ limiting	multitag	field_on_ control	field_on_ time

7.22.5 RF170C

Nachfolgend werden die Openness-Parameter für die Baugruppe "RF170C" beschrieben.

Tabelle 7-12 Parameter der Baugruppe "RF170C"; Parameter der Parametergruppe "Baugruppenparameter"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
User Mode	UserMode	ulong	Ident-Profil/ RFID-Normprofil	3	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter wählen Sie den Baustein aus: • Ident-Profil/RFID-Normprofil: In der Steuerung kommt der Programmbaustein für das Ident-Profil/RFID-Normprofil zum Einsatz. • FB 45 / FC 45: Singletag-Betrieb. In der Steuerung wird der FB 45 (PROFIBUS/PROFINET) oder der FC 45 (PROFIBUS) eingesetzt. • FB 55 / FC 55: Multitag-Betrieb. In der Steuerung wird der FB 55 (PROFIBUS/PROFINET) oder der FC 55 (PROFIBUS) eingesetzt.
			FB 45 / FC 45	1	R/W	
			FB 55 / FC 55	4	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
MOBY Mode	MobyMode	ulong	RF200/ RF300/ RF600; MV4x0; MOBY U/D	5	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Betriebsart des Kommunikationsmoduls ein. • RF200/RF300/RF600; MV4x0; MOBY U/D • MOBY I/E • MV3xx • Freeport-Protokoll • RF300 Filehandler Normaladressierung: Der Transponder wird mit physikalischen Adressen angesprochen. Filehandler: Der Transponder muss vor der Benutzung formatiert werden.
			MOBY I/E	1	R/W	
			MV3xx	16	R/W	
			Freeport-Protokoll	17	R/W	
			RF300 Filehandler	149	R/W	
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Diagnosemeldungen	DiagnosticMessages	int	Keine	1	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie ein, ob Hardware-Diagnosemeldungen gemeldet werden sollen. • Keine: Neben der Standarddiagnose werden keine weiteren Alarme generiert. • Hard Errors: Schwerwiegende Hardware-Fehler werden über die S7-Diagnose gemeldet.
			Hard Errors	2	R/W	
Unterdrückung Error-LED	SuppressionOfErrorLed	int	Keine	0	R/W	Deaktivieren der Error-LED (ERR) eines Kanals. Das Kommunikationsmodul verfügt über zwei Kanäle, an die Reader / optischen Lesegeräte angeschlossen werden können. Wird nur einer der Kanäle verwendet, blinkt die Error-LED des anderen Kanals permanent. Mithilfe der Unterdrückung können Sie die Error-LED des nicht verwendeten Kanals deaktivieren.
			Kanal 1	1	R/W	
			Kanal 2	2	R/W	
Schnittstelle	ModuleInterface	int	RS232	1	R	Auswahl des Schnittstellen-Typs, den die angeschlossene Hardware (Reader / optische Lesegeräte) verwendet.
			RS422	0	R	

Tabelle 7-13 Parameter der Baugruppe "RF170C"; Parameter der Parametergruppe "Freeport-Protokoll"

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Datenbits	DataBits	ulong	7	7	R/W	Auswahl der Anzahl Bits, auf die ein Zeichen abgebildet wird.
			8	8	R/W	

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Parität	Parity	ulong	Keine	0	R/W	Auswahl der Parität Eine Folge von Datenbits kann um ein Paritätsbit erweitert werden. Das Paritätsbit ergänzt durch seinen Wert "0" oder "1" die Summe aller Bits (Datenbits und Paritätsbits) auf einen definierten Zustand. Die Datensicherheit wird dadurch erhöht. • Keine: Daten werden ohne Paritätsbit gesendet. • Ungerade: Paritätsbit wird so gesetzt, dass die Summe der Datenbits (inkl. Paritätsbit) mit Signalzustand "1" ungerade ist. • Gerade: Paritätsbit wird so gesetzt, dass die Summe der Datenbits (inkl. Paritätsbit) mit Signalzustand "1" gerade ist. • Fixwert 1: Paritätsbit wird fest auf den Wert "1" gesetzt. • Fixwert 0: Paritätsbit wird fest auf den Wert "0" gesetzt.
			Ungerade	1	R/W	
			Gerade	2	R/W	
			Fixwert 1	3	R/W	
			Fixwert 0	7	R/W	
Stopbits	StopBits	ulong	1	1	R/W	Auswahl der Anzahl Stopbits, die das Ende eines Zeichens kennzeichnen. Die Stopbits werden bei der Übertragung an jedes übertragene Zeichen angehängt.
			2	2	R/W	

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
Festlegung der Endeerkennung	EndDetectionOfAReceivedFrame	ulong	Nach Ablauf der Zeichenverzugszeit	16	R/W	Festlegung der Endeerkennung eines Empfangstelegramms auf: • Nach Ablauf der Zeichenverzugszeit: Das Telegramm hat weder eine feste Läge noch definierte Endezeichen. Das Ende eines Telegramms ist durch eine Lücke in der Zeichenfolge bestimmt. Die Größe dieser Lücke wird durch die Zeichenverzugszeit festgelegt. • Nach Empfang einer festen Zeichenanzahl: Die Läge der Empfangstelegramme ist immer gleich. Beim Empfang von Daten wird das Telegrammende erkannt, wenn die parametrierte Anzahl von Zeichen empfangen wird. • Nach Empfang der/des Endezeichen(s): Am Ende des Telegramms stehen ein oder zwei definierte Endezeichen. Beim Empfang von Daten wird das Telegrammende erkannt, wenn das parametrierte Endezeichen empfangen wird
			Nach Empfang einer festen Zeichenanzahl	32	R/W	
			Nach Empfang der/des Endezeichen(s)	49	R/W	
Anzahl der Endezeichen	NumberOfEndDelimiters	ulong	1	1	R/W	Auswahl der Anzahl der Endezeichen. Es können maximal 2 Endezeichen projektiert werden. Beim Empfang von Daten wird das Telegrammende erkannt, wenn die gewählte Endezeichenkombination empfangen wird.
			2	2	R/W	

UI-Parameter	Openness-Parameter	Datentyp in Openness	UI-Parameterwert	Openess-Parameterwert	Default-Zugriff	Beschreibung
1. Endezeichen	FirstEndDelimiterReceiver	ulong	0...7F / 0...FF	<text>	R/W	Eingabe des 1. Endezeichens von maximal zwei Endezeichen für Endekriterium "Nach Empfang der/des Endezeichen(s)". Das gewählte Endezeichen bzw. die gewählteEndezeichenkombination begrenzt die Länge des jeweiligen Telegramms. Parameterwert abhängig vom Parameter "Datenbits".
2. Endezeichen	SecondEndDelimiterReceiver	ulong	0...7F / 0...FF	<text>	R	Eingabe des 2. Endezeichens von maximal zwei Endezeichen für Endekriterium "Nach Empfang der/des Endezeichen(s)". Das gewählte Endezeichen bzw. die gewählte Endezeichenkombination begrenzt die Länge des jeweiligen Telegramms. Parameterwert abhängig vom Parameter "Datenbits".
Telegrammlänge	FrameLength	ulong	1...233 / 1...229	<text>	R	Eingabe der Telegrammlänge in Byte für das Endekriterium "Nach Empfang einer festen Zeichenanzahl".
Zeichenverzugszeit	CharacterDelayTime	ulong	0...65535	<text>	R	Eingabe der Zeit [ms], die verstreichen darf, bis ein Telegrammende erkannt wird. Wählen Sie die Zeichenverzugszeit in Abhängigkeit vom Sendeverhalten Ihres Kommunikationspartners. Die Zeichenverzugszeit ist in Abhängigkeit von der Datenübertragungsgeschwindigkeit auf einen Mindestwert begrenzt. Beachten Sie, dass der ASCII-Treiber auch beim Senden eine Pause zwischen zwei Telegrammen einhält.

7.22.6 RF180C

Nachfolgend werden die Openness-Parameter für die Baugruppe "RF180C" beschrieben.

Tabelle 7-14 Parameter der Baugruppe "RF180C"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
User Mode	UserMode	ulong	FB 45	1	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter wählen Sie den Baustein aus: • FB 45: Singletag-Betrieb. In der Steuerung wird der FB 45 (PROFIBUS/ PROFINET)eingesetzt. • FB 55: Multitag-Betrieb. In der Steuerung wird der FB 55 (PROFIBUS/PROFINET) eingesetzt. • FB 56: Multitag-Betrieb. In der Steuerung wird der FB 56 (PROFIBUS/PROFINET) eingesetzt. • RFID-Normprofil: In der Steuerung kommt der Programmbaustein für das RFID-Normprofil zum Einsatz.
			FB 55	4	R/W	
			FB 56	5	R/W	
			RFID-Normprofil	3	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
MOBY Mode	MobyMode	ulong	MOBY I/E Normaladressierung	1	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Betriebsart des Kommunikationsmoduls ein. • MOBY I/E-Normaladressierung • MOBY I-Filehandler • RF200/RF300/RF600; MOBY U/D-Normaladr. • MOBY U-Filehandler • RF300 Filehandler Normaladressierung: Der Transponder wird mit physikalischen Adressen angesprochen. Filehandler: Der Transponder muss vor der Benutzung formatiert werden.
			MOBY I Filehandler	129	R/W	
			RF200/ RF300/ RF600; MV4x0; MOBY U/D-Normaladr.	5	R/W	
			MOBY U Filehandler	133	R/W	
			RF300 Filehandler	149	R/W	
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Diagnosemeldungen	DiagnosticMessages	int	Keine	1	R/W	Mit diesem Parameter stellen Sie ein, ob Hardware-Diagnosemeldungen gemeldet werden sollen. • Keine: Neben der Standarddiagnose werden keine weiteren Alarme generiert. • Hard Errors: Schwerwiegende Hardware-Fehler werden über die S7-Diagnose gemeldet. • Hard/Soft Errors: Schwerwiegende Hardware-Fehler, sowie Fehler, die während der Befehlsbearbeitung auftreten, werden über die S7-Diagnose gemeldet.
			Hard Errors	2	R/W	
			Hard/Soft Errors	4	R/W	
Unterdrückung Error-LED	SuppressionOfErrorLed	int	Keine	0	R/W	Deaktivieren der Error-LED (ERR) eines Kanals. Das Kommunikationsmodul verfügt über zwei Kanäle, an die Reader / optischen Lesegeräte angeschlossen werden können. Wird nur einer der Kanäle verwendet, blinkt die Error-LED des anderen Kanals permanent. Mithilfe der Unterdrückung können Sie die Error-LED des nicht verwendeten Kanals deaktivieren.
			Kanal 1	1	R/W	
			Kanal 2	2	R/W	

7.22.7 RF18xC

Nachfolgend werden die Openness-Parameter für die Baugruppen "RF18xC" beschrieben.

Tabelle 7-15 Parameter der Baugruppen "RF18xC"; Parameter der Parametergruppe "Grundparameter"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
User Mode	UserMode	ulong	Ident-Profil/ RFID-Normprofil	3	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter wählen Sie den Baustein aus: • Ident-Profil/RFID-Normprofil: In der Steuerung kommt der Programmbaustein für das Ident-Profil/RFID-Normprofil zum Einsatz.

Tabelle 7-16 Parameter der Baugruppen "RF18xC"; Parameter der Parametergruppe "Baugruppenparameter"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
User Mode	UserMode	ulong	FB 45	1	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter wählen Sie den Baustein aus: • FB 45: Singletag-Betrieb. In der Steuerung wird der FB 45 (PROFIBUS/ PROFINET)eingesetzt. • Ident-Profil/RFID-Normprofil: In der Steuerung kommt der Programmbaustein für das Ident-Profil/RFID-Normprofil zum Einsatz.
			Ident-Profil/ RFID-Normprofil	3	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
MOBY Mode	MobyMode	ulong	RF200/ RF300/ RF600; MV4x0; MOBY U/D-Normaladr.	5	R	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Betriebsart des Kommunikationsmoduls ein. • RF200/RF300/RF600; MOBY U/D-Normaladr. Normaladressierung: Der Transponder wird mit physikalischen Adressen angesprochen. Filehandler: Der Transponder muss vor der Benutzung formatiert werden.
Übertragungsgeschwindigkeit	BaudRate	ulong	19,2 kBd	9	R/W	Die Auswahl ist abhängig vom verwendeten Kommunikationsmodul und Ident-System. Mit diesem Parameter stellen Sie die Datenübertragungsgeschwindigkeit zwischen Kommunikationsmodul und Reader ein. Bei angeschlossenem RFID-Reader: Nach der Umstellung der Übertragungsgeschwindigkeit muss der Reader aus- und wieder eingeschaltet werden (Spannung aus/ein). Bei angeschlossenem optischen Lesegerät: Die hier ausgewählte Übertragungsgeschwindigkeit muss mit der in der Firmware des Lesegerätes ausgewählten Übertragungsgeschwindigkeit übereinstimmen.
			57,6 kBd	13	R/W	
			115,2 kBd	14	R/W	

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Diagnosemeldungen	Diagnostic-Messages	int	Keine	1	R/W	Mit diesem Parameter stellen Sie ein, ob Hardware-Diagnosemeldungen gemeldet werden sollen. • Keine: Neben der Standarddiagnose werden keine weiteren Alarme generiert. • Hard Errors: Schwerwiegende Hardware-Fehler werden über die S7-Diagnose gemeldet. • Hard/Soft Errors: Schwerwiegende Hardware-Fehler, sowie Fehler, die während der Befehlsbearbeitung auftreten, werden über die S7-Diagnose gemeldet.
			Hard Errors	2	R/W	
			Hard/Soft Errors	4	R/W	

7.22.8 RF615R/RF680R/RF685R

Nachfolgend werden die Openness-Parameter für die Baugruppen "RF615R/RF680R/RF685R" beschrieben.

Tabelle 7-17 Parameter der Baugruppen "RF615R/RF680R/RF685R"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
RFID Lesestellen-Alarm 1	RfidReadPointAlarm1	int	Aus	0	R/W	Lesestellebezogene Diagnosemeldungen ein-/ausschalten.
			An	1	R/W	
RFID Lesestellen-Alarm 2	RfidReadPointAlarm2	int	Aus	0	R/W	
			An	1	R/W	
RFID Lesestellen-Alarm 3 [1]	RfidReadPointAlarm3	int	Aus	0	R/W	
			An	1	R/W	
RFID Lesestellen-Alarm 4 [1]	RfidReadPointAlarm4	int	Aus	0	R/W	
			An	1	R/W	

[1] Ausschließlich bei RF680R

7.22.9 MV400/MV500

Nachfolgend werden die Openness-Parameter für die Baugruppen "MV400/MV500" beschrieben.

Tabelle 7-18 Parameter der Baugruppen "MV400/MV500"

Parameter in der Konfiguration (TIA Portal)	Parameter in Openness	Datentyp in Openness	Parameterwert in der Konfiguration (TIA Portal)	Parameterwert in Openness	Default-Zugriff	Beschreibung
Funktionsbaustein	FunctionBlock	int	FB79	0	R/W	Mit diesem Parameter wählen Sie den Baustein aus: • FB79 Kompatibel zu dem Lesegerät "VS130-2"; Kompatible Steuerungen: S7-300 und S7-400 • Ident-Profil: Komplexer Ident-Baustein mit MV-spezifischen Bausteinen; Kompatible Steuerungen: S7-300, S7-400, S7-1200 und S7-1500
			Ident-Profil	1	R/W	

7.23 Ausnahmen

7.23.1 Umgang mit Exceptions

Ausnahmen beim Zugriff auf das TIA Portal über TIA Portal Openness APIs

Bei der Ausführung einer TIA Portal Openness-Anwendung über die TIA Portal Openness API werden alle auftretenden Fehler als Ausnahmen gemeldet. Diese Ausnahmen enthalten Informationen, die Ihnen helfen, die aufgetretenen Fehler zu beheben.

Dabei wird zwischen zwei Arten von Ausnahmen unterschieden:

- Recoverable (Siemens.Engineering.EngineeringException)
  Sie können mit dieser Ausnahme weiterhin ohne Unterbrechung auf das TIA Portal zugreifen. Alternativ können Sie die Verbindung zum TIA Portal abbrechen.
  Die EngineeringExceptions beinhalten die folgenden Typen:
  - Sicherheitsbezogene Ausnahmen (EngineeringSecurityException), beispielsweise bei fehlenden Zugriffsrechten.
  - Ausnahmen beim Zugriff auf Objekte (EngineeringObjectDisposedException), beispielsweise beim Zugriff auf nicht mehr vorhandene Objekte.
  - Ausnahmen beim Zugriff auf Attribute (EngineeringNotSupportedException), beispielsweise beim Zugriff auf nicht mehr vorhandene Attribute.
  - Allgemeine Ausnahmen beim Aufruf (EngineeringTargetInvocationException), beispielsweise bei einem Fehler trotz gültigen Aufrufs der TIA Portal Openness API.
  - Ausnahmen beim Aufruf (EngineeringRuntimeException), beispielsweise bei einer ungültigen Belegung.
  - Ausnahmen, wenn es nicht genügend Ressourcen in der zugeordneten TIA Portal-Instanz gibt (EngineeringOutOfMemoryException)
  - Ausnahmen bei beendeten Aufrufen (EngineeringUserAbortException), beispielsweise beim Abbruch des Importvorgangs durch den Benutzer.
  - Ausnahmen beim API-Aufruf durch einen vom Kunden bereitgestellten Delegate (EngineeringDelegateInvocationException). Diese Ausnahme wird von der Ausnahme EngineeringTargetInvocationException abgeleitet.

  Die EngineeringExceptions haben die folgenden Attribute:
  - `ExceptionMessageData messageData`: Enthält den Grund, weshalb die Ausnahme ausgelöst wurde.
  - `ExceptionMessageData detailMessageData`: Enthält zusätzliche Informationen über den Grund. Das Ergebnis wird als <IList> zurückgegeben.
  - `String message`: Gibt das Ergebnis von `MessageData` und `DetailMessageData` zurück.

  `ExceptionMessageData` gibt die folgenden Informationen zurück:
  - `String Text`: Enthält den Grund, weshalb die Ausnahme ausgelöst wurde.
- NonRecoverable (Siemens.Engineering.NonRecoverableException)
  Diese Ausnahme schließt das TIA Portal und die Verbindung zum TIA Portal wird getrennt. Sie müssen das TIA Portal mit der TIA Portal Openness-Anwendung erneut starten.

Programmcode

Das folgende Beispiel zeigt die Möglichkeiten, die Sie haben, um auf Ausnahmen zu reagieren:

```
try
{
    ...
}

catch(EngineeringSecurityException engineeringSecurityException)
{
    Console.WriteLine(engineeringSecurityException);
}

catch(EngineeringObjectDisposedException engineeringObjectDisposedException)
{
    Console.WriteLine(engineeringObjectDisposedException.Message);
}

catch(EngineeringNotSupportedException engineeringNotSupportedException)
{
    Console.WriteLine(engineeringNotSupportedException.MessageData.Text);
    Console.WriteLine();
    foreach(ExceptionMessageData detailMessageData in
engineeringNotSupportedException.DetailMessageData)
    {
        Console.WriteLine(detailMessageData.Text);
    }
}

catch (EngineeringTargetInvocationException)
{
    throw;
}

catch (EngineeringException)
{
    //Do not catch general exceptions
    throw;
}

catch(NonRecoverableException nonRecoverableException)
{
    Console.WriteLine(nonRecoverableException.Message);
}
```

Export/Import

8

8.1 Überblick

8.1.1 Grundlagen zum Import/Export

Einleitung

Sie können bestimmte Konfigurationsdaten exportieren und die Daten nach dem Editieren in dasselbe oder ein anderes Projekt reimportieren.

Hinweis

Im Zusammenhang mit dieser Beschreibung gibt es keinerlei Verpflichtungen oder Garantien, die Quelldatei manuell zu ändern und auszuwerten. Siemens übernimmt deshalb keine Haftung für den Gebrauch dieser Beschreibung als Ganzes oder in Teilen.

Exportierbare und importierbare Objekte

Die folgenden Konfigurationsdaten können auch mit TIA Portal Openness APIs importiert oder exportiert werden:

Tabelle 8-1 Projekte

Objekte	Export	Import
Grafiksammlung	X	X

Tabelle 8-2 PLC

Objekte	Export	Import
Bausteine	X	X
Knowhow-geschützte Bausteine	X	–
Fehlersichere Bausteine	X	–
Systembausteine	X	–
PLC-Variablentabellen	X	X
PLC-Variablen und -Konstanten	X	X
Anwenderdatentypen	X	X

Tabelle 8-3 HMI

Objekte	Export	Import
Bilder	X	X
Bildvorlagen	X	X
Globale Bilder	X	X
Pop-up-Bilder	X	X
Slide-in-Bilder	X	X
Skripte	X	X
Textlisten	X	X
Grafiklisten	X	X
Zyklen	X	X
Verbindungen	X	X
Variablentabelle	X	X
Variablen	X	X

Vollständiger Export oder Export offener Referenzen

Die oben aufgeführten Objekttypen werden mit allen Objekten exportiert oder importiert, wenn diese demselben Unterbaum angehören. Diese Regel gilt auch für referenzierte Objekte im selben Unterbaum.

Bei referenzierten Objekten in anderen Unterbäumen ist jedoch ein vollständiger Export oder Import nicht möglich. Stattdessen werden "offene Referenzen" zu diesen Objekten exportiert oder importiert.

Referenzierte Objekte aus demselben Unterbaum werden nur exportiert, wenn sie der Gruppe exportierbarer Objekte angehören. Alle Dynamisierungen an Objekten werden beim Importieren/Exportieren als Objekte behandelt und ebenfalls exportiert/importiert.

Der Export umfasst alle Objektattribute, die bei der Konfiguration geändert wurden. Das gilt unabhängig davon, ob das geänderte Attribut genutzt wird oder nicht.

Beispiel: Sie haben ein grafisches EA-Feld mit dem Modus "Eingabe/Ausgabe" konfiguriert und die Einstellung "Sichtbar nach Anklicken" für das Attribut "Rollbalken" ausgewählt. Im Laufe der Konfiguration haben Sie den Modus in "Zwei Zustände" geändert. Das Attribut "Rollbalken" ist in diesem Modus nicht verfügbar. Weil das Attribut "Rollbalken" geändert wurde, wird es in den Export einbezogen, selbst wenn das Attribut nicht genutzt wird.

Geöffnete Referenzen importieren

Sie können auch Objekte mit geöffneten Referenzen importieren (siehe Import von Projektierungsdaten (Seite 429)).

Wenn die referenzierten Objekte im Zielprojekt enthalten sind, werden die geöffneten Referenzen automatisch wieder mit den Objekttypen verknüpft. Diese Objekte müssen am gleichen Ort verfügbar sein und dem gleichen Namen wie für den Export zugeordnet sein. Wenn die referenzierten Objekte nicht im Zielprojekt enthalten sind, können die geöffneten

Referenzen nicht aufgelöst werden. Es wird kein zusätzliches Objekt erzeugt, um diese geöffneten Referenzen aufzulösen.

Dateiformat exportieren und importieren

Das Dateiformat zum Exportieren und Importieren ist XML. Nur CAx-Daten erfordern das AML-Format. Die unterschiedlichen Schemadefinitionen für sämtliche Formate werden in dem jeweiligen Abschnitt in diesem Handbuch beschrieben:

- XML-Format für die Daten eines HMI-Geräts (Seite 437)
- XML-Format für die Daten eines PLC-Geräts (Seite 489)
- AML-Format für CAx-Daten (Seite 562)

Schriftarten importieren und exportieren

Für Objekte definierte Schriftarten werden ebenfalls exportiert und importiert.

Wenn Sie nicht im Projekt enthaltene Schriftarten importieren, wird nach dem Import die Standardschriftart am Objekt angezeigt. Jedoch wird die importierte Schriftart in der Datenverwaltung gespeichert.

Wenn die Attribute für eine Schriftart nicht in einer Importdatei zuwiesen sind, werden die Attribute nach dem Import mit Standardwerten belegt.

Einschränkungen

Das Exportformat ist intern und ausschließlich für die aktuelle Version von TIA Portal Openness gültig. Das Exportformat kann sich in künftigen Versionen ändern.

Alle beim Import und Export auftretenden Fehler werden als Ausnahmen gemeldet. Weitere Informationen über Ausnahmen siehe Abschnitt Umgang mit Exceptions (Seite 420).

Siehe auch

Einsatzgebiet von Import/Export (Seite 425)

Export von Projektierungsdaten (Seite 427)

8.1.2 Einsatzgebiet von Import/Export

Einleitung

Die Funktionalität zum Importieren/Exportieren ermöglicht Ihnen, bestimmte Objekte gezielt zu exportieren.

Sie können die exportierten Daten in einem externen Programm editieren oder unverändert in anderen TIA-Portal-Projekten wiederverwenden.

Wenn Sie die Importdatei richtig strukturieren, können Sie auch extern erzeugte Konfigurationsdaten importieren, ohne zuerst einen Export durchführen zu müssen.

Hinweis

Wenn Sie extern erzeugte Konfigurationsdaten mit Codefehlern oder falscher Struktur importieren, kann dies unerwartete Fehler verursachen.

Anwendungsbereich

Exportieren und Importieren von Daten ist nützlich für folgende Aufgaben:

- Externes Editieren von Konfigurationsdaten.
- Importieren von extern erzeugten Konfigurationsdaten, z. B. Textlisten und Variablen.
- Verteilen von spezifizierten Konfigurationsdaten an verschiedene Projekte, z. B. ein geändertes Prozessbild, das in mehreren Projekten verwendet werden soll.
- Zum Replizieren und Anpassen der Hardware-Konfiguration zwischen dem TIA-Portal-Projekt und einem ECAD-Programm.

Siehe auch

Grundlagen zum Import/Export (Seite 423)

8.1.3 Versionsspezifischer SimaticML-Import

Anwendung

Ab TIA Portal Openness V14 SP1 ist der SimaticML-Import versionsübergreifend verwendbar. Sie können Ihre älteren Exportdateien mindestens in die nächsten zwei größeren Versionen importieren.

Jede Version der Openness API kann SIMATIC-ML-Dateien aus der entsprechenden Version und aus jeder unterstützten Version des vorherigen Release importieren; beispielsweise wird der Import von SIMATIC-ML-Dateien V14 SP1 in Openness API V15.1 unterstützt.

Die folgende Tabelle zeigt ein Beispiel dafür, welche SIMATIC-ML-Version von welcher Openness API-Version importiert werden kann.

	SIMATIC-ML-Datei V14 SP1	SIMATIC-ML-Datei V15	SIMATIC-ML-Datei V15.1
Openness API V14 SP1	Import unterstützt	Import nicht unterstützt	Import nicht unterstützt
Openness API V15 SP1	Import unterstützt	Import unterstützt	Import nicht unterstützt
Openness API V15.1	Import unterstützt	Import unterstützt	Import unterstützt

Die einzelnen Versionen der Openness API unterstützen den Export von SIMATIC-ML-Dateien. Die Version der exportierten SIMATIC-ML-Datei sollte jedoch mit der Version des TIA Portals und nicht mit der Version der verwendeten Openness API übereinstimmen.

Um dieses Merkmal zu unterstützen, enthalten SimaticML-Dateien jetzt die Modellversionsinformationen wie nachstehend dargestellt:

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
    <Engineering version="V14 SP1"/>
    <DocumentInfo>
     ...
    </DocumentInfo>
    <SW.DataBlock ID="0">
     ...
    </SW.DataBlock>
</Document>
```

Hinweis

Wenn die Versionsinformationen nicht in der SimaticML-Datei enthalten sind, verwendet das System die aktuelle Modellversion.

8.1.4 Bearbeiten der XML-Datei

Einleitung

Zur Bearbeitung einer XML-Datei für den Import von Projektierungsdaten verwenden Sie einen XML-Editor oder einen Texteditor.

Wenn Sie umfangreiche Änderungen durchführen oder selbst Objektstrukturen aufbauen, empfehlen wir die Verwendung eines XML-Editors mit Auto-Vervollständigungsfunktion.

Hinweis

Ändern des XML-Inhalts setzt umfangreiche Kenntnisse der Struktur und der Validierungsregeln in XML voraus. Vermeiden Sie Validierungsfehler und arbeiten Sie nur in Ausnahmefällen manuell in der XML-Struktur.

8.1.5 Export von Projektierungsdaten

Einleitung

Die Konfigurationsdaten jedes Startobjekts (Stamm) werden separat in eine XML-Datei exportiert.

Zum Editieren der Exportdatei sind angemessene Kenntnisse von XML erforderlich. Verwenden Sie für komfortables Editieren einen XML-Editor.

Beispiel

Sie haben ein Prozessbild, das ein EA-Feld enthält. In dem EA-Feld ist eine externe Variable konfiguriert. Ein Export des Prozessbilds enthält das Bild und das EA-Feld. Die Variable und die von der Variable verwendete Verbindung werden nicht exportiert. Stattdessen wird nur eine offene Referenz in den Export einbezogen.

Inhalt der Exportdatei

Beginnend mit dem Startobjekt werden alle Objekte eines Unterbaums und deren Attribute in der Exportdatei gespeichert. Alle Verweise auf Objekte anderer Unterbäume werden nur als offene Verweise exportiert. Die entsprechenden Attribute der referenzierten Objekte in verschiedenen Unterbäumen werden nicht in die Exportdatei geschrieben.

Hinweis

Der Export von Objekttypen aus der Bibliothek wird nicht unterstützt.

Sie können Objekte als Typ in der Bibliothek erzeugen. Instanzen des im Projekt verwendeten Objekttyps können wie andere Objekte mit der TIA Portal Openness-Anwendung bearbeitet werden. Wenn Sie Objekte exportieren, werden die Instanzen ohne Typinformationen exportiert.

Wenn Sie diese Objekte wieder in das Projekt importieren, werden die Instanzen des Objekttyps überschrieben und die Instanz wird vom Objekttyp abgelöst.

Die Exportdatei enthält nicht notwendigerweise alle Attribute eines Objekts. Sie legen fest, welche Daten exportiert werden sollen:

- `ExportOptions.None`
  Diese Einstellung exportiert nur die geänderten Daten oder die Daten, die von den Standardwerten abweichen.
  Die Exportdatei enthält auch alle Werte, die für den anschließenden Datenimport obligatorisch sind.
- `ExportOptions.WithDefaults`[1]
  Auch die Standardwerte werden exportiert.
- `ExportOptions.WithReadOnly`[1]
  Auch die schreibgeschützten Werte werden exportiert.

[1]: Sie können diese beiden Optionen mit der folgenden Syntax kombinieren:

```
Export(path,ExportOptions.WithDefaults |
ExportOptions.WithReadOnly);
```

Sämtliche Inhalte der Exportdatei liegen auf Englisch vor. Unabhängig davon werden enthaltene Projekttexte in alle vorhandenen Sprachen exportiert und importiert.

Alle Konfigurationsdaten werden als XML-Objekte in der Exportdatei modelliert.

Siehe auch

Grundlagen zum Import/Export (Seite 423)

Bausteine exportieren (Seite 501)

8.1.6 Import von Projektierungsdaten

Einleitung

Konfigurationsdaten werden aus einer zuvor exportierten und editierten XML-Datei oder aus einer von Ihnen selbst erzeugten XML-Datei importiert. Die in dieser Datei enthaltenen Daten werden beim Import geprüft. Dieser Ansatz verhindert, dass die Konfigurationsdaten im TIA Portal infolge des Imports inkonsistent werden.

Einschränkungen

- Alle Stammobjekte in der Importdatei müssen vom gleichen Typ sein, z. B. Variablentabellen, Bausteine usw.
- Wenn eine Importdatei mehrere Stammobjekte enthält und eines davon nicht gültig ist, werden sämtliche Inhalte der Importdatei nicht importiert.
- Beim Importieren von Texten müssen die entsprechenden Projektsprachen im Zielprojekt eingerichtet worden sein, um ein Fehlschlagen des Imports zu vermeiden. Bei Bedarf können Sie die Spracheinstellungen über TIA Portal Openness ändern.
- Wenn Sie ungültige Attribute eines Objekts in der Importdatei angeben, die nicht in der grafischen Benutzeroberfläche des TIA Portals editiert werden können, wird der Import abgebrochen.
- Nur die im Feld "separately for each connection" aufgeführten Bereichszeiger können importiert oder exportiert werden.
- Der Import von Objekttypen aus der Bibliothek wird nicht unterstützt. Sie können Objekte als Typ in der Bibliothek erzeugen. Instanzen des im Projekt verwendeten Objekttyps können wie andere Objekte mit der TIA Portal Openness-Anwendung bearbeitet werden. Wenn Sie Objekte exportieren, werden die Instanzen ohne Typinformationen exportiert. Wenn Sie diese Objekte wieder in das Projekt importieren, werden die Instanzen des Objekttyps überschrieben und die Instanz wird vom Objekttyp abgelöst.
- Der Import fehlersicherer Bausteine wird nicht unterstützt.

Hinweis

Geräteabhängige Wertebereiche für grafische Attribute

Wenn Werte grafischer Attribute außerhalb des gültigen Wertebereichs liegen, werden diese Werte beim Import auf die möglichen Maximalwerte für das HMI-Gerät zurückgesetzt.

Unterschiedliches Importverhalten

Wenn bereits im Projekt vorhandene Objekte importiert werden sollen, müssen Sie das Importverhalten mit verschiedenen Programmcodes steuern. Andernfalls werden die Objekte beim Import wieder im Projekt erzeugt.

Die folgenden Einstellungen für das Importverhalten sind möglich:

- `ImportOptions.None`
  Bei dieser Einstellung werden die Konfigurationsdaten ohne Überschreiben importiert. Wenn ein Objekt aus einer bereits im Projekt vorhandenen XML-Datei importiert wird, wird der Import unterbrochen und eine Ausnahme ausgelöst.
- `ImportOptions.Override`
  Bei Verwendung dieser Einstellung werden die Konfigurationsdaten mit automatischem Überschreiben importiert.
  Sie können angeben, dass im Projekt vorhandene Objekte beim Import überschrieben werden sollen. Relevante Objekte werden vor dem Import gelöscht und mit Standardwerten neu erzeugt. Diese Standardwerte werden beim Import mit den importierten Werten überschrieben. Wenn das vorhandene Objekt und das neue Objekt nicht in derselben Gruppe sind, kann kein Überschreiben stattfinden. Um Namenskonflikte zu vermeiden, wird der Import abgebrochen und eine Ausnahme ausgelöst.

Vorgehensweise zum Importieren

Wenn Sie eine XML-Datei importieren möchten, müssen die darin enthaltenen Daten bestimmten Regeln entsprechen. Die Inhalte der Importdatei müssen wohlgeformt sein. Es dürfen keine Syntaxfehler und keine Datenstrukturfehler vorhanden sein. Verwenden Sie bei umfangreichen Änderungen einen XML-Editor, der diese Kriterien vor dem Import überprüft.

Beim Import der XML-Datei in das TIA Portal werden die Daten in der Datei zuerst auf formale Fehler im XML-Code geprüft. Wenn bei dieser Prüfung Fehler erkannt werden, wird der Import abgebrochen und die Fehler werden in einer Ausnahme angezeigt (siehe Umgang mit Exceptions (Seite 420)).

Siehe auch

Grundlagen zum Import/Export (Seite 423)

Anwenderdatentyp importieren (Seite 559)

8.2 Import/Export von Projektdaten

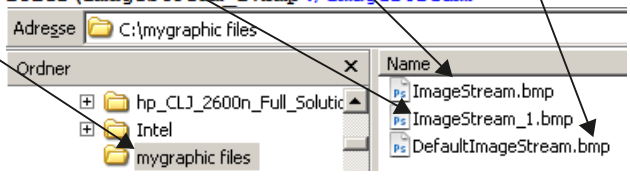
8.2.1 Grafiksammlung

8.2.1.1 Export/Import von Grafiken

Einleitung

Der Export von Konfigurationsdaten aus dem TIA Portal in die XML-Datei beinhaltet keine ausgewählten Grafiken oder Grafiken, auf die durch ein Objekt verwiesen wird. Die Grafiken werden beim Export separat gespeichert. In der XML-Datei wird anhand eines relativen Pfads und die Dateinamen auf die Grafiken verwiesen. Eine Grafikreferenz wird in der XML-Datei als ein Objekt modelliert und enthält eine Attributliste und bei Bedarf eine Verknüpfungsliste wie bei den anderen Objekten.

```
<Hmi.Globalization.MultiLingualGraphic ID="0">
    <AttributeList>
      <DefaultDithering>False</DefaultDithering>
      <DefaultImageStream external="path">mygraphic files\DefaultImageStream.bmp</DefaultImageStream>
      <DefaultSmoothness>False</DefaultSmoothness>
      <Name>MyGraphic1</Name>
    </AttributeList>
    <ObjectList>
      <Hmi.Globalization.GraphicItem ID="1" CompositionName="Items">
        <AttributeList>
          <Culture>en-US</Culture>
          <Dithering>False</Dithering>
          <ImageStream external="path">mygraphic files\ImageStream.bmp</ImageStream>
          <Smoothness>False</Smoothness>
        </AttributeList>
      </Hmi.Globalization.GraphicItem>
      <Hmi.Globalization.GraphicItem ID="2" CompositionName="Items">
        <AttributeList>
          <Culture>de-DE</Culture>
          <Dithering>False</Dithering>
          <ImageStream external="path">mygraphic files\ImageStream_1.bmp</ImageStream>
          <Smoothness>False</Smoothness>
        </AttributeList>
      </Hmi.Globalization.GraphicItem>
    </ObjectList>
  </Hmi.Globalization.MultiLingualGraphic>
```



Grafiken exportieren

Der Export von Konfigurationsdaten beinhaltet ausschließlich Grafiken, die für den Export direkt ausgewählt wurden. Die exportierbaren Grafiken werden für die jeweilige Sprache im TIA Portal gespeichert. Wird ein Projekt in mehreren Sprachen konfiguriert, dann werden alle verwendeten Sprachversionen exportiert.

Beim Grafikexport wird ein neuer Ordner im Exportdateiordner erstellt. Der Dateiordnername wird durch Verknüpfen des XML-Dateinamens mit "Dateien" aufgebaut. Dieser Ordner enthält die exportierten Grafiken. Falls dieser Ordner bereits vorhanden ist, wird ein neuer Ordner erstellt und durch eine laufende Nummer ergänzt.

Die Grafiken werden in demselben Dateiformat wie im Projekt gespeichert. Das Datenformat wird nicht geändert oder konvertiert und die Auflösung und Farbtiefe bleiben unverändert.

Die Kennung "default" wird als Dateierweiterung für die als Standardsprache ausgewählte Sprache verwendet.

Wenn der Ordner bereits eine Datei mit demselben Namen enthält, wird der Dateiname der exportierten Grafik durch eine laufende Nummer ergänzt.

Grafiken importieren

Beim Importieren von Grafiken gelten die folgenden Voraussetzungen:

- Die Grafiken müssen ein vom TIA Portal unterstütztes Dateiformat besitzen.
- Auf die Grafiken muss in der XML-Datei durch Angabe eines relativen Pfads verwiesen werden.

Nach dem Export einer Grafik kann diese mithilfe eines Grafikprogramms außerhalb des TIA Portals bearbeitet und anschließend wieder importiert werden.

Siehe auch

Grundlagen zum Import/Export (Seite 423)

8.2.1.2 Grafiken eines Projektes exportieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Sie können entweder eine einzelne Grafik oder alle Grafiken der Grafiksammlung eines Projekts in allen Sprachen exportieren. Während des Exports wird eine XML-Datei mit allen betroffenen Projektgrafikeinträgen erstellt und zusammen mit den exportierten Grafiken referenziert. Die entsprechenden Grafiken werden zusammen mit der XML-Datei im selben Verzeichnis des Dateisystems gespeichert.

Damit die exportierten Grafiken ("*.jpg", "*.bmp", "*.png", "*.ico", etc.) geändert werden können, sind diese Grafiken nicht schreibgeschützt.

Programmcode: Grafik exportieren

Um die erforderliche Grafik zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all language variants of a single grafic
Project project = …;
MultiLingualGraphicComposition graphicsComposition = project.Graphics;
MultiLingualGraphic graphic = graphicsComposition.Find("graphicName");
graphic.Export(new FileInfo(@"D:\ExportFolder\graphicName.xml"),
ExportOptions.WithDefaults);
```

Programmcode: Alle Grafiken exportieren

Um alle Grafiken einer Grafiksammlung zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all graphics of a graphic library
Project project = …;
MultiLingualGraphicComposition graphicsComposition = project.Graphics;
foreach(MultiLingualGraphic graphic in graphicsComposition)
{
    graphic.Export(new FileInfo(string.Format(@"D:\Graphics\{0}.xml", graphic.Name)),
ExportOptions.WithDefaults);
}
```

8.2.1.3 Grafiken in ein Projekt importieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Eine XML-Datei wird zusammen mit den Sprachversionen einer Grafik in einem Verzeichnis Ihres Dateisystems gespeichert.

Sie können alle Grafiken in einem relativen Pfad in der XML-Datei referenzieren.

Sie können jetzt alle Sprachversionen einer in der XML-Datei enthaltenen Grafik in die Grafiksammlung importieren.

Beachten Sie auch Folgendes: Import von Projektierungsdaten (Seite 429).

Programmcode

Um eine oder mehrere Grafiken zu importieren, ändern Sie den folgenden Programmcode:

```
//Import all language variants of a single graphic
Project project = …;
MultiLingualGraphicComposition graphicComposition = project.Graphics;
graphicComposition.Import(new FileInfo(@"D:\Graphics\Graphic1.xml"),
ImportOptions.Override);
```

8.2.2 Projekttexte

8.2.2.1 Export von Projekttexten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Im TIA Portal finden Sie Projekttexte unter dem Knoten "Sprachen & Ressourcen" eines Projekts. Diese Texte werden in eine *.xlsx-Datei exportiert, die zum Beispiel für Übersetzungen genutzt werden kann. Die Einschränkungen beim Exportieren und Importieren von Projekttexten sind die gleichen wie in der Benutzeroberfläche. Diese Einschränkungen umfassen:

- Exportierte Texte können nur in das Projekt importiert werden, aus dem sie exportiert wurde.
- Texte können nur in Sprachen übersetzt werden, die im Projekt verfügbar sind. Bei Bedarf können Sie Projektsprachen über TIA Portal Openness hinzufügen.
- Nur vorhandene Texte können reimportiert werden. Wenn Texte aus dem ursprünglichen Projekt gelöscht oder neu erzeugt wurden, schlägt der Import dieser Texte fehl.

Sie müssen die folgenden Parameter festlegen:

Name	Beispiel	Beschreibung
pah	new FileInfo ("D:\Test\Project-Text.xlsx")	Pfad zur Exportdatei
sourceLanguage	new CultureInfo("en-US")	Referenzsprache, aus der Text zu übersetzen ist
targetLanguage	new CultureInfo("de-DE")	Zielsprache, in die Text zu übersetzen ist

Hinweis

Mehrsprachige Texte werden zusammen mit dem übergeordneten Objekt, zu dem sie gehören, exportiert. Mehrsprachige Texte können nicht explizit exportiert werden.

Programmcode: Aus dem Knoten "Sprachen & Ressourcen" exportieren

Die Verwendung der Beispielparameter führt zum folgenden Programmcode zum Exportieren von Projekttexten:

```
project.ExportProjectTexts(new FileInfo(@"D:\Test\ProjectText.xlsx"), new CultureInfo("en-US"), new CultureInfo("de-DE"));
```

XML-Struktur eines exportierten mehrsprachigen Textelements

```
...
  <MultilingualText ID="2" CompositionName="Comment">
    <ObjectList>
      <MultilingualTextItem ID="3" CompositionName="Items">
        <AttributeList>
          <Culture>en-US</Culture>
          <Text>My super tag</Text>
        </AttributeList>
      </MultilingualTextItem>
      <MultilingualTextItem ID="4" CompositionName="Items">
        <AttributeList>
          <Culture>ru-RU</Culture>
          <Text>Мой супер тэг</Text>
        </AttributeList>
      </MultilingualTextItem>
    </ObjectList>
  </MultilingualText>
...
```

8.2.2.2 Import von Projekttexten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Im TIA Portal finden Sie Projekttexte unter dem Knoten "Sprachen & Ressourcen" eines Projekts. Sie können Projekttexte aus einer *.xlsx-Datei importieren, was zum Beispiel für Übersetzungen genutzt werden kann. Die Einschränkungen beim Exportieren und Importieren von Projekttexten sind die gleichen wie in der Benutzeroberfläche. Diese Einschränkungen umfassen:

- Exportierte Texte können nur in das Projekt importiert werden, aus dem sie exportiert wurde.
- Übersetzte Texte können nur in Sprachen importiert werden, die in dem Projekt, aus dem sie exportiert wurden, verfügbar sind.
- Nur vorhandene Texte können reimportiert werden. Wenn Texte aus dem ursprünglichen Projekt gelöscht oder neu erzeugt wurden, schlägt der Import dieser Texte fehl.

Sie müssen die folgenden Parameter festlegen:

Name	Beispiel	Beschreibung
path	new FileInfo(@"D:\Test\Project-Text.xlsx")	Pfad zur Importdatei
updateSourceLanguage	true	Wenn "true", wird der Text der Referenzsprache anhand der Exportdatei aktualisiert. Wenn "false", wird der Text der Referenzsprache nicht aktualisiert.

Hinweis

Mehrsprachige Texte werden zusammen mit dem übergeordneten Objekt, zu dem sie gehören, importiert. Mehrsprachige Texte können nicht explizit importiert werden.

Programmcode

Die Verwendung der Beispielparameter führt zum folgenden Programmcode zum Importieren von Projekttexten:

```
ProjectTextResult result = project.ImportProjectTexts(new FileInfo(@"D:\Test
\ProjectText.xlsx"), true);
```

Der Import der Projekttexte gibt ein Objekt zurück, das den Status des Imports anzeigt und den Pfad angibt, in dem das Importprotokoll gespeichert wird. Auf diese Attribute kann mit dem folgenden Code zugegriffen werden:

```
ProjectTextResultState resultState = result.State;
FileInfo logFilePath = result.Path;
```

8.3 Import/Export von Daten eines HMI-Geräts

8.3.1 Aufbau einer XML-Datei

Einleitung

Die Daten in der Exportdatei vom Import/Export sind mit Verweis auf eine Grundstruktur gegliedert.

Grundstruktur einer Exportdatei

Die Exportdatei wird in einem XML-Format erzeugt.

Die XML-Datei beginnt mit einer Dokumentinformation. Sie beinhaltet die Daten der rechnerspezifischen Installation, mit der das Projekt exportiert wurde.

Die Exportdatei ist in die folgenden zwei Abschnitte unterteilt:

- Informationen über das Dokument
  In diesem Abschnitt können Sie Ihre eigenen Informationen über den Export in gültiger XML-Syntax eingeben. Der Inhalt wird beim Import ignoriert.
  Zum Beispiel können Sie einen Baustein mit `<IntegrityInformation>...</IntegrityInformation>` einfügen und darin zusätzliche Informationen über die Validierung unterbringen. Nach Weiterleitung der XML-Datei kann der Empfänger anhand dieses Bausteins vor dem Import prüfen, ob die XML-Datei geändert wurde.
- Objekt
  Dieser Abschnitt enthält die zu exportierenden Elemente.

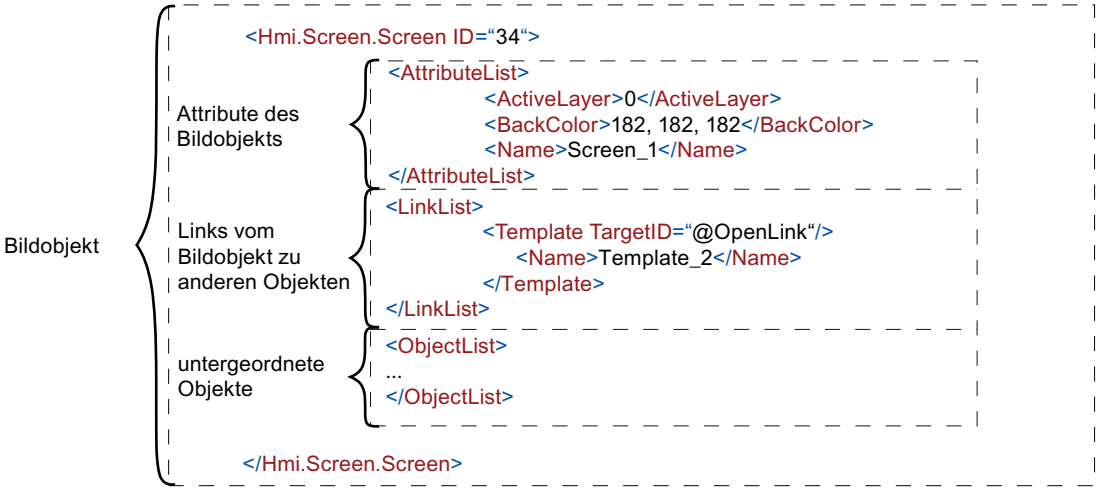
Informationen über das Dokument

Bildobjekt

```
<?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>
```

Bildobjekte einer Exportdatei

Die exportierten Elemente sind in zusätzlichen Elementen der XML-Datei verfügbar.



Siehe auch

Grundlagen zum Import/Export (Seite 423)

8.3.2 Struktur der Daten für Import/Export

Objekte

Die Grundstruktur ist für alle Objekte gleich.

Jedes Objekt in der XML-Datei beginnt mit seinem Typ, zum Beispiel "Hmi.Screen.Button", und einer ID. Die ID wird beim Export automatisch erzeugt.

```
<Hmi.Screen.Button CompositionName="ScreenItems" ID="60">
```

Jedes Objekt mit Ausnahme des Startobjekts enthält auch ein XML-Attribut "CompositionName". Der Wert für dieses Attribut ist voreingestellt. Es ist gelegentlich notwendig, dieses Attribut anzugeben, zum Beispiel zum Ändern der Beschriftung beim Drücken oder Loslassen einer Schaltfläche.

```
<MultilingualText ID="A" CompositionName="TextOff">
    <ObjectList>
        <MultilingualTextItem ID="B" CompositionName="Items">
            <AttributeList>
                <Culture>en-US</Culture>
                <Text>
                    <body>
                        <p>TextOff</p>
                    </body>
                </Text>
            </AttributeList>
        </MultilingualTextItem>
    </ObjectList>
</MultilingualText>
<MultilingualText ID="C" CompositionName="TextOn">
    <ObjectList>
        <MultilingualTextItem ID="D" CompositionName="Items">
            <AttributeList>
                <Culture>en-US</Culture>
                <Text>
                    <body>
                        <p>TextOn</p>
                    </body>
                </Text>
            </AttributeList>
        </MultilingualTextItem>
    </ObjectList>
</MultilingualText>
```

Attribute

Jedes Objekt enthält Attribute, die in einem Abschnitt "AttributeList" enthalten sind. Jedes Attribut ist als XML-Element modelliert, z. B. "BackColor". Der Wert eines Attributs ist als XML-Inhalt modelliert, z. B. "204, 204, 204".

```
<Hmi.Screen.Button ID="2" CompositionName="ScreenItems">
    <AttributeList>
        <BackColor>204, 204, 204</BackColor>
        <ObjectName>Button_1</ObjectName>
    </AttributeList>
</Hmi.Screen.Button>
```

Bei referenzierenden Objekten enthält jedes Objekt gegebenenfalls einen Abschnitt "LinkList". Dieser Abschnitt enthält Verknüpfungen mit anderen Objekten innerhalb oder außerhalb der XML-Datei. Jede Verknüpfung ist als XML-Element modelliert. Die Bezeichnung einer Verknüpfung wird vom Zielobjekt in der Schemadatei festgelegt. Jede Verknüpfung enthält auch das Attribut "TargetID". Wenn das Zielobjekt in der XML-Datei enthalten ist, ist der Wert des Attributs "TargetID" die ID des referenzierten Objekts mit vorangestellter Raute "#". Wenn das Zielobjekt nicht in der XML-Datei enthalten ist, hat das Attribut "TargetID" den Wert "@OpenLink". Die tatsächliche Referenz des Objekts ist als untergeordnetes XML-Element modelliert.

```
<Hmi.Tag.Tag ID="17">
    <AttributeList>
        <Name>Tag_1</Name>
    </AttributeList>
    <LinkList>
        <AcquisitionCycle TargetID="@OpenLink">
            <Name>2 s</Name>
        </AcquisitionCycle>
        <Connection TargetID="@OpenLink">
            <Name>HMI_connection</Name>
        </Connection>
    </LinkList>
</Hmi.Tag.Tag>
```

Beziehung zwischen Objekten und XML-Struktur

Die nachstehenden Bilder zeigen die Beziehung zwischen der exportierten XML-Struktur und den zugeordneten Objekten in WinCC.

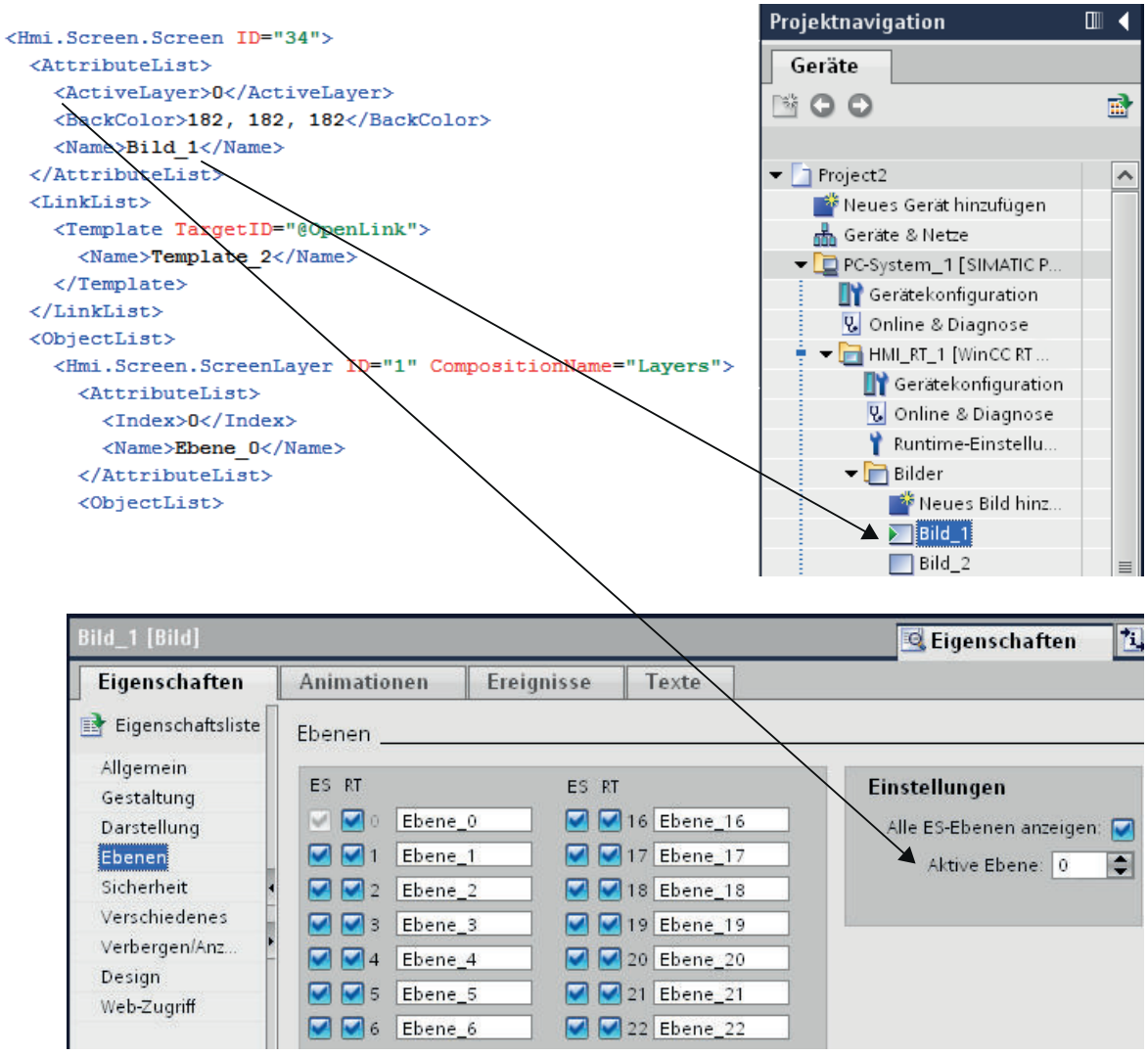


Bild 8-1 Beziehung zwischen WinCC-Benutzeroberfläche und XML-Struktur.

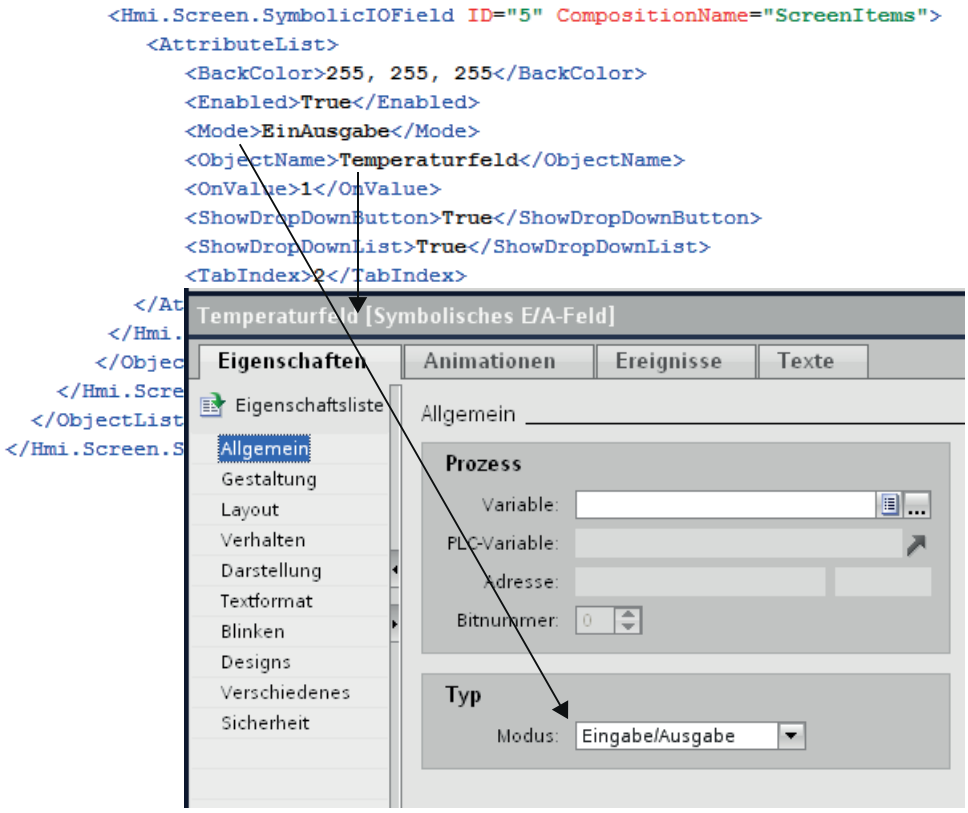


Bild 8-2 Beziehung zwischen den Einstellungen in WinCC und XML-Struktur.

8.3.3 Zyklen

8.3.3.1 Zyklen exportieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die Schnittstelle TIA Portal Openness API unterstützt den Export aller Zyklen eines bekannten HMI-Geräts in eine XML-Datei. Die Generierung der entsprechenden Exportdatei weist darauf hin, dass der Export abgeschlossen ist.

Programmcode

Um Zyklen aus einem HMI-Gerät in eine XML-Datei zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports cycles from an HMI device
private static void ExportCyclesFromHMITarget(HmiTarget hmitarget)
{
    CycleComposition cycles = hmitarget.Cycles;
    foreach(Cycle cycle in cycles)
       {
           cycle.Export(new FileInfo(string.Format(@"C:\Samples\{0}.xml", cycle.Name)),
ExportOptions.WithDefaults);
       }
}
```

8.3.3.2 Zyklen importieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Wenn Sie `ImportOptions.None` verwenden, können Sie anhand der Zusammensetzungszahl (Composition count) die Zyklen ermitteln, die tatsächlich importiert wurden. Sie haben Zugriff auf diese importierten Zyklen.

Hinweis

Standardzyklen haben Attribute, die in der Benutzeroberfläche nicht bearbeitet werden können. Wenn Sie in der Importdatei angeben, dass diese Attribute geändert werden sollen, verursacht der Importvorgang eine NonRecoverableException und schließt das TIA Portal.

Programmcode

Um einen Zyklus oder mehrere Zyklen aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports cycles to an HMI device
private static void ImportCyclesToHMITarget(HmiTarget hmitarget)
{
    CycleComposition cycles = hmitarget.Cycles;
    string dirPathImport = @"C:\OpennessSamples\Import\";
    string cycleImportFileName = "CycleImport.xml";
    string fullFilePath = Path.Combine(dirPathImport, cycleImportFileName);

    cycles.Import(new FileInfo(fullFilePath), ImportOptions.None);
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

8.3.4 Variablentabellen

8.3.4.1 HMI-Variablentabellen exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Verwendung

Pro HMI-Variablentabelle wird eine XML-Datei exportiert. Die API unterstützt diesen Exportvorgang. Der Export von Variablentabellen ist auch in Unterordnern verfügbar.

Programmcode: Alle HMI-Variablentabellen aus einem angegebenen Ordner exportieren

Um alle HMI-Variablentabellen aus einem bestimmten Ordner zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all tag tables from a tag folder
private static void ExportAllTagTablesFromTagFolder(HmiTarget hmitarget)
{
    TagSystemFolder folder = hmitarget.TagFolder;
    TagTableComposition tables = folder.TagTables;

    foreach (TagTable table in tables)
    {
       FileInfo info = new FileInfo(string.Format(@"C:\OpennessSamples\TagTables\{0}.xml",
table.Name));
        table.Export(info, ExportOptions.WithDefaults);
    }
}
```

Programmcode: Eine HMI-Variablentabelle exportieren

Um eine einzelne HMI-Variablentabelle zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports a tag table from an HMI device
private static void ExportTagTableFromHMITarget(HmiTarget hmitarget)
{
    string tableName = "Tag table XYZ";
    TagSystemFolder folder = hmitarget.TagFolder;
    TagTableComposition tables = folder.TagTables;
    TagTable table = tables.Find(tableName);

    if (table != null)
    {
       FileInfo info = new FileInfo(string.Format(@"C:\OpennessSamples\TagTables\{0}.xml",
table.Name));
        table.Export(info, ExportOptions.WithDefaults);
    }
}
```

Programmcode: Alle HMI-Variablentabellen exportieren

Um alle HMI-Variablentabellen zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all tag tables from an HMI device
private static void ExportAllTagTablesFromHMITarget(HmiTarget hmitarget)
{
    TagSystemFolder sysFolder = hmitarget.TagFolder;

    //First export the tables in underlying user folder
    foreach (TagUserFolder userFolder in sysFolder.Folders)
    {
        ExportUserFolderDeep(userFolder);
    }

    //then, export all tables in the system folder
    ExportTablesInSystemFolder(sysFolder);
}

private static void ExportUserFolderDeep(TagUserFolder rootUserFolder)
{
        foreach (TagUserFolder userFolder in rootUserFolder.Folders)
        {
            ExportUserFolderDeep(userFolder);
        }
        ExportTablesInUserFolder(rootUserFolder);
}

private static void ExportTablesInUserFolder(TagUserFolder folderToExport)
{
     TagTableComposition tables = folderToExport.TagTables;
     foreach (TagTable table in tables)
     {
         string fullFilePath = string.Format(@"C:\OpennessSamples\TagTables\{0}.xml",
table.Name);
         table.Export(new FileInfo(fullFilePath), ExportOptions.WithDefaults);
     }
 }

private static void ExportTablesInSystemFolder(TagSystemFolder folderToExport)
{
     TagTableComposition tables = folderToExport.TagTables;
     foreach (TagTable table in tables)
     {
         string fullFilePath = string.Format(@"C:\OpennessSamples\TagTables\{0}.xml",
table.Name);
         table.Export(new FileInfo(fullFilePath), ExportOptions.WithDefaults);
     }
 }
```

8.3.4.2 HMI-Variablentabelle importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Programmcode

Um die HMI-Variablentabelle einer XML-Datei in einen benutzerdefinierten Ordner oder in einen Systemordner zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports a single HMI tag table from a XML file
private static void ImportSingleHMITagTable(HmiTarget hmitarget)
{
    TagSystemFolder folder = hmitarget.TagFolder;
    TagTableComposition tables = folder.TagTables;

    FileInfo info = new FileInfo(@"D:\Samples\Import\myExportedTagTable.xml");
    tables.Import(info, ImportOptions.Override);
}
```

Fehlerhafter Import von Variablen

Wenn Sie in den Namen von Variablen oder referenzierten Variablen die folgenden Symbole verwenden, kommt es beim Import der Variablen zu Fehlern:

- . (Punkt)
- \ (Backslash)

Abhilfe 1:

Vergewissern Sie sich vor einem Export, dass der Name der zu exportierenden Variable oder Bezugsvariable keinen Punkt und keinen Backslash enthält.

Abhilfe 2:

Schließen Sie in der Importdatei die Namen von Variablen oder referenzierten Variablen mit Anführungszeichen aus.

Beispiel

- Variablenname mit Symbol:
  <name>Siemens.Simatic.Hmi.Utah.Tag.HmiTag:41000_Options_Time_Date \DB_SFX0908_HMI1.Actual_Date_Time.Hour</name>
- Variablenname mit ausgeschlossenem Symbol in Anführungszeichen:
  <name>"Siemens.Simatic.Hmi.Utah.Tag.HmiTag:41000_Options_Time_Date \DB_SFX0908_HMI1.Actual_Date_Time.Hour"</name>

8.3.4.3 Einzelne Variable einer HMI-Variablentabelle exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Verwendung

Die folgenden Objekttypen des Objektmodells können als untergeordnete Elemente einer HMI-Variable vorhanden sein und werden beim Export berücksichtigt:

MultilingualText	Für Kommentar, TagValue, DisplayName
TagArrayMemberTag	Für HMI-Array-Elemente
TagStructureMember	Für HMI-Strukturelemente
Event	Für konfigurierte Ereignisse
MultiplexEntry	Für konfigurierte Multiplex-Einträge von Variablen

Programmcode

Um eine einzelne Variable aus einer HMI-Variablentabelle in eine XML-Datei zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports a selected tag from a tag table
private static void ExportSelectedTagFromTagTable(HmiTarget hmitarget)
{
    TagSystemFolder tagFolder = hmitarget.TagFolder;
    TagTable mytable = tagFolder.TagTables.Find("MyTagTable");

    TagComposition containingTags = mytable.Tags;
    Tag myTag = containingTags.Find("MyTag");

    if (myTag != null)
    {
        FileInfo info = new FileInfo(string.Format(@"C:\OpennessSamples\Tags\{0}.xml",
myTag.Name));
        myTag.Export(info, ExportOptions.WithDefaults);
    }
}
```

8.3.4.4 Einzelne Variable in eine HMI-Variablentabelle importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Verwendung

Die folgenden Objektmodell-Objekttypen können als untergeordnete Elemente einer HMI-Variable vorhanden sein und werden beim Import berücksichtigt:

MultilingualText	Für Kommentar, TagValue, DisplayName
TagArrayMemberTag	Für HMI-Array-Elemente
TagStructureMember	Für HMI-Strukturelemente
Event	Für konfigurierte Ereignisse
MultiplexEntry	Für konfigurierte Multiplex-Einträge von Variablen

Programmcode

Um eine HMI-Variable aus einer XML-Datei in eine HMI-Variablentabelle zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports a tag into a tag table
private static void ImportTagIntoTagTable(HmiTarget hmitarget)
{
    TagSystemFolder tagFolder = hmitarget.TagFolder;
    TagTable myTable = tagFolder.DefaultTagTable;
    TagComposition tagComposition = myTable.Tags;

    FileInfo info = new FileInfo(@"D:\Samples\Import\myExportedTag.xml");
    tagComposition.Import(info, ImportOptions.Override);
}
```

8.3.4.5 Besonderheiten beim Export/Import von HMI-Variablen

Einleitung

Besondere Überlegungen gelten beim Export und Import der folgenden HMI-Variablen:

- Externe HMI-Variablen mit integrierter Verbindung
- HMI-Variablen mit dem Datentyp "UDT"

Ähnliche Programmcodes

Der Programmcode für die oben genannten HMI-Variablen ist fast identisch mit den folgenden Programmcodes:

- Programmcode: HMI-Variablen exportieren (Seite 449)
- Programmcode: HMI-Variablen importieren (Seite 450)

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Besondere Überlegungen für den Export/Import einer externen HMI-Variable mit integrierter Verbindung

Beim Exportieren einer externen HMI-Variable mit integrierter HMI-Verbindung wird statt der PLC-Variablendaten nur die Verknüpfung der HMI-Variable mit der PLC-Variable in der Exportdatei gespeichert.

Vor dem Import müssen Sie sicherstellen, dass der PLC, die entsprechenden PLC-Variablen und die integrierte Verbindung zum entsprechenden PLC im Projekt vorhanden sind. Wenn das nicht der Fall ist, müssen diese Elemente vor dem Import erzeugt werden. Beim anschließenden Import der externen HMI-Variable wird die Verknüpfung zur PLC-Variable wieder aktiviert.

Namen externer HMI-Variablen müssen über alle Variablentabellen eines Projekts eindeutig sein. Wenn Sie beim Import nicht die passende Variablentabelle für die HMI-Variable angeben, wird der Import abgebrochen.

Verwenden Sie zum Importieren einer externen HMI-Variable mit integrierter Verbindung die folgende XML-Struktur:

```
<Hmi.Tag.Tag ID="1" CompositionName="Tags">
    <AttributeList>
        <Name>MyIntegratedHmiTag_1</Name>
    </AttributeList>
    <LinkList>
        <AcquisitionCycle TargetID="@OpenLink">
            <Name>1 s</Name>
        </AcquisitionCycle>
        <Connection TargetID="@OpenLink">
            <Name>HMI_Connection_MP277_300400</Name>    <- Must exist in the project
        </Connection>
        <ControllerTag TargetID="@OpenLink">
            <Name>Datablock_1.DBElement1</Name>        <- Must exist in the project
        </ControllerTag>
    </LinkList>
</Hmi.Tag.Tag>
```

Besondere Überlegungen für den Export/Import einer HMI-Variable des Datentyps "UDT"

Die Verknüpfung wird zum Datentyp exportiert, wenn eine HMI-Variable des Datentyps "UDT" exportiert wird. Nur versionierte Datentypen werden für den Import unterstützt.

Die Datentypen müssen in der Projektbibliothek gespeichert werden. Datentypen in der globalen Bibliothek werden nicht unterstützt.

Die folgenden Regeln gelten beim Import:

- Der referenzierte Datentyp muss in der Projektbibliothek enthalten sein.
  Der Import wird beendet, wenn der Datentyp nicht in der Projektbibliothek enthalten ist.
- Der referenzierte Datentyp muss versioniert sein. Versionierung wird ab TIA Portal V13 SP1 unterstützt.
  Wenn der Datentyp nicht versioniert ist, wird eine Ausnahme ausgelöst.

Hinweis

Der erste gefundene Datentyp wird beim Import zum Auflösen der Referenz verwendet.

Hierbei gilt Folgendes: Zuerst wird das Wurzelverzeichnis der Projektbibliothek durchsucht, danach die Unterordner.

Verwenden Sie die folgende XML-Struktur, um eine HMI-Variable des Datentyps "UDT" zu importieren:

```
<Hmi.Tag.Tag ID="1" CompositionName="Tags">
    ...
    <LinkList>
        <DataType TargetID="@OpenLink">
            <Name>HmiUdt_1 V 1.0.0</Name>    <- Must exist in the project library
        </DataType>
        ...
    </LinkList>
    ...
</Hmi.Tag.Tag>
```

8.3.5 VB-Skripte

8.3.5.1 VB-Skripte exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Alle untergeordneten benutzerdefinierten Ordner werden beim Export berücksichtigt. Für jedes exportierte VB-Skript wird eine eigene XML-Datei erstellt.

Programmcode: VB-Skript exportieren

Um ein ausgewähltes VB-Skript eines HMI-Geräts in eine XML-Datei zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports a single vbscript of an HMI device
private static void ExportSingleVBScriptOfHMITarget(HmiTarget hmitarget)
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptComposition vbScripts = vbScriptFolder.VBScripts;
    VBScript vbScript = vbScripts.Find("MyVBScript");

    FileInfo info = new FileInfo(string.Format(@"C:\OpennessSamples\Export\Scripts
\{0}.xml", vbScript.Name));
    vbScript.Export(info, ExportOptions.None);
}
```

8.3.5.2 VB-Skripte aus einem Ordner exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Für jedes exportierte VB-Skript wird eine eigene XML-Datei erstellt.

Programmcode: VB-Skript aus einem benutzerdefinierten Ordner exportieren

Um ein VB-Skript aus einem benutzerdefinierten Ordner in eine XML-Datei zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports vbscripts of a selected vbscript system folder
private static void ExportVBScriptOfSelectedVBScriptSystemFolder(HmiTarget hmitarget)
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptUserFolderComposition vbUserFolders = vbScriptFolder.Folders;
    VBScriptUserFolder vbUserFolder = vbUserFolders.Find("MyVBUserFolder");
    VBScriptComposition vbScripts = vbUserFolder.VBScripts;

    foreach (VBScript script in vbScripts)
    {
        FileInfo info = new FileInfo(String.Format(@"C:\OpennessSamples\Export\Scripts
\{0}.xml", script.Name));
        script.Export(info, ExportOptions.None);
    }
}
```

Programmcode: Alle VB-Skripte aus einem Systemordner exportieren

Um alle VB-Skripte aus dem Systemordner zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports all vbscripts by using a foreach loop
private static void ExportAllVBScripts(HmiTarget hmitarget)
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptComposition vbScripts = vbScriptFolder.VBScripts;
    if (vbScripts == null) return;

    foreach (VBScript script in vbScripts)
    {
        FileInfo info = new FileInfo(string.Format(@"C:\OpennessSamples\Export\Scripts
\{0}.xml", script.Name));
        script.Export(info, ExportOptions.None);
    }
}
```

8.3.5.3 VB-Skripte importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Sammelimporte werden unterstützt. Als Alternative können Sie einen Programmcode mit einer Foreach-Schleife verwenden (VB-Skripte exportieren (Seite 452)).

Programmcode

Um ein VB-Skript aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie den folgenden Programmcode:

```
private static void ImportSingleVBScriptToHMITarget(HmiTarget hmitarget)
{
    VBScriptSystemFolder vbScriptFolder = hmitarget.VBScriptFolder;
    VBScriptComposition vbScripts = vbScriptFolder.VBScripts;
    if (vbScripts 00 null) return;
    {
        FileInfo info = new FileInfo(@"D:\Samples\Import\VBScript.xml");
        vbScripts.Import(info, ImportOptions.None);
    }
}
```

8.3.6 Textlisten

8.3.6.1 Textlisten aus einem HMI-Gerät exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Der Export von Text- und Grafiklisten umfasst alle Einträge der Listen. Text- und Grafiklisten können getrennt exportiert werden.

Die Textlisten eines HMI-Geräts werden exportiert. Für jede exportierte Textliste wird eine eigene XML-Datei erstellt.

Programmcode

Ändern Sie den folgenden Programmcode, um Textlisten aus einem HMI-Gerät zu exportieren:

```
//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 Textliste in ein HMI-Gerät importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die API-Schnittstelle unterstützt den Import einer Textliste aus einer XML-Datei in ein HMI-Gerät.

Programmcode

Um eine Textliste aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports a single TextList
private static void ImportSingleTextList(HmiTarget hmitarget)
{
    TextListComposition textListComposition = hmitarget.TextLists;
    IList<TextList> importedTextLists = textListComposition.Import(new FileInfo(@"D:
\SamplesImport\myTextList.xml"), ImportOptions.Override);
}
```

8.3.6.3 Erweiterte XML-Formate für Export/Import von Textlisten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Standardexport von Textlisten
  Siehe Textlisten aus einem Bediengerät exportieren (Seite 455)
- Standardimport von Textlisten
  Siehe Textlisten in ein Bediengerät exportieren (Seite 456)

Verwendung

Eine Textliste kann auch formatierte Texte enthalten. Das betrifft hauptsächlich die folgende Formatierung:

- Textformatierung
- Verweise auf andere Objekte im Text

Reine Textformatierung in einer zu exportierenden Textliste führt zu einem erweiterten XML-Exportformat. Objektreferenzen werden als Open Links charakterisiert. Gleiches gilt für zu importierende Textlisten mit formatierten Texten.

Erweiterte XML-Exportformate können auch erheblich komplexer werden. Beispielsweise kann mehr als nur der Objektname in der Textliste verknüpft sein, möglicherweise durch ein Open Link zu einer PLC-Variable eines anderen Geräts. In diesem Fall müssen alle Informationen in einem String codiert werden, um das Open Link zu entfernen.

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
<!-- ... -->
    <MultilingualText ID="5" CompositionName="Text">
        <ObjectList>
            <MultilingualTextItem ID="6" CompositionName="Items">
                <AttributeList>
                    <Culture>en-US</Culture>
                    <Text>
                        <body>
                            <p>
                                <field ref="0" />
                            </p>
                        </body>
                        <fieldinfos>
                            <fieldinfo name="0" domaintype="HMICommonTextList">
                                <reference TargetID="@OpenLink">
                                    <name>Siemens.Simatic.Hmi.Utah.TextAndGraphicLists.HmiTextList:Empty Text_list_
                                </reference>
                                <subreference TargetID="@OpenLink">
                                    <name>Siemens.Simatic.Hmi.Utah.Tag.HmiTag:t1</name>
                                </subreference>
                                <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_list_
                                </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 Grafiklisten

8.3.7.1 Grafiklisten exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Der Export von Text- und Grafiklisten umfasst alle Einträge der Listen. Text- und Grafiklisten können getrennt exportiert werden.

Pro Grafikliste wird eine XML-Datei erstellt. In den Grafiklisten enthaltene globale Grafikobjekte werden als Open Links exportiert.

Programmcode

Um Grafiklisten aus einem Bediengerät zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports GraphicLists
private static void ExportGraphicLists(HmiTarget hmitarget)
{
    GraphicListComposition graphic = hmitarget.GraphicLists;
    foreach (GraphicList graphicList in graphic)
    {
        FileInfo info = new FileInfo(string.Format(@"D:\Samples\Export\{0}.xml",
graphicList.Name));
        graphicList.Export(info, ExportOptions.WithDefaults);
    }
}
```

8.3.7.2 Grafikliste importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die API-Schnittstelle unterstützt den Import einer Grafikliste aus einer XML-Datei in ein HMI-Gerät.

Alle referenzierten Grafikobjekte der Grafikliste werden in den Import einbezogen. Referenzen auf globale Grafiken werden nicht einbezogen. Wenn die referenzierten globalen Grafiken im Zielprojekt vorhanden sind, werden die Referenzen auf die globalen Grafiken während des Imports wiederhergestellt.

Programmcode

Um eine Grafikliste aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
//Imports a single GraphicList
private static void ImportSingleGraphicList(HmiTarget hmitarget)
{
    GraphicListComposition graphicListComposition = hmitarget.GraphicLists;
    IList<GraphicList> importedGraphicLists = graphicListComposition.Import(new
FileInfo(@"D:\Samples\Import\myGraphicList.xml"), ImportOptions.Override);
}
```

8.3.8 Verbindungen

8.3.8.1 Verbindungen exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Verwendung

Die API-Schnittstelle unterstützt den Export aller Verbindungen eines HMI-Geräts in eine XML-Datei.

Hinweis

Integrierte Verbindungen exportieren

Der Export integrierter Verbindungen wird nicht unterstützt.

Für jede exportierte Verbindung wird eine eigene XML-Datei erstellt.

Programmcode

Um alle Verbindungen eines HMI-Geräts in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports communication connections from an HMI device
private static void ExportConnectionsFromHMITarget(HmiTarget hmitarget)
{
    ConnectionComposition connections = hmitarget.Connections;
    foreach(Connection connection in connections)
    {
        FileInfo info = new FileInfo(string.Format(@"D:\Samples\Export\{0}.xml",
connection.Name));
        connextion.Export(info, ExportOptions.WithDefaults);
    }
}
```

8.3.8.2 Verbindungen importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Verwendung

Die API-Schnittstelle unterstützt den Import aller Verbindungen eines HMI-Geräts aus einer XML-Datei in ein HMI-Gerät. Wenn Sie mehrere Kommunikationsverbindungen importieren möchten, importieren Sie die jeweilige XML-Datei für die entsprechende Verbindung.

Hinweis

Wenn Sie eine Verbindung in ein Projekt importieren, in dem bereits eine integrierte Verbindung konfiguriert ist, wird diese Verbindung nicht überschrieben. Der Import wird abgebrochen und eine Exception wird ausgelöst.

Programmcode

Um eine einzelne Verbindung eines HMI-Geräts aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
//Imports Communication connections to an HMI device
private static void ImportConnectionsToHMITarget(HmiTarget hmitarget)
{
    ConnectionComposition connections = hmitarget.Connections;
    IList<Connection> importedConnectionLists = connections.Import(new FileInfo(@"D:
\Samples\Import\myConnectionImport.xml"), ImportOptions.Override);
}
```

8.3.9 Bilder

8.3.9.1 Übersicht der exportierbaren Bild-Objekte

Verwendung

Sie können die nachstehenden Bilder mit TIA Portal Openness APIs exportieren und importieren:

Tabelle 8-4 Unterstützte Bilder

Objekt	Export/Import möglich
Bild	Ja
Globales Bild	Ja
Bildvorlage	Ja
Permanentfenster	Ja
Pop-up-Bild	Ja
Slide-in-Bild	Ja

Sie können die nachstehenden Bildobjekte mit TIA Portal Openness APIs exportieren und importieren:

Tabelle 8-5 Unterstützte Bildobjekte

Bereich	Objekttyp	Export/Import möglich
Grundlegende Objekte	Linie	Ja
	Polygonzug	Ja
	Polygon	Ja
	Ellipse	Ja
	Ellipsensegment	–
	Kreissegment	–
	Ellipsenbogen	–
	Kreisbogen	–
	Kreis	Ja
	Rechteck	Ja
	Verbinder	–
	Textfeld	Ja
	Grafikanzeige	Ja
	Rohr	–
	Doppel-T-Stück	–
	T-Stück	–
	Rohrkrümmer	–

Bereich	Objekttyp	Export/Import möglich
Elemente	EA-Feld	Ja
	Grafisches EA-Feld	Ja
	Editierbares Textfeld	–
	Listenfeld	–
	Kombinationsfeld	–
	Schaltfläche	Ja
	Rundbutton	–
	Leuchtdrucktaster	Ja
	Schalter	Ja
	Symbolisches EA-Feld	Ja
	Datum/Uhrzeit-Feld	Ja
	Balken	Ja
	Symbolbibliothek	Ja
	Schieberegler	Ja
	Bildlaufleiste	–
	Kontrollkästchen	–
	Optionsschaltfläche	–
	Zeigerinstrument	Ja
	Uhr	Ja
	Speicherplatzanzeige	–
	Funktionstasten (Softkeys)	Ja
	Gruppen	Ja
	Bildbausteininstanzen	Ja

Bereich	Objekttyp	Export/Import möglich
Bedienelemente	Bildfenster	–
	Benutzeranzeige	Ja
	Druckauftrag/Skriptdiagnose	–
	Kamera-Anzeige	–
	PDF-Anzeige	–
	Rezepturanzeige	–
	Meldeanzeige	–
	Meldeindikator	–
	Meldefenster	–
	f(x)-Kurvenanzeige	–
	f(t)-Kurvenanzeige	–
	Tabellenanzeige	–
	Wertetabelle	–
	HTML-Browser	–
	Media Player	–
	Kanaldiagnose	–
	WLAN-Empfang	–
	Zonen-Name	–
	Zonen-Signal	–
	Wirkbereichs-Name	–
	Wirkbereichs-Name (RFID)	–
	Wirkbereichs-Signal	–
	Ladezustand	–
	Handrad	–
	Hilfe-Indikator	–
	Sm@rtClient-Anzeige	–
	Status/Steuern	–
	Speicherplatzanzeige	–
	Anzeige von NC-Unterprogrammen	–
	Systemdiagnoseanzeige	–
	System-Diagnosefenster	–

Siehe auch

Grundlagen zum Import/Export (Seite 423)

8.3.9.2 Alle Bilder eines HMI-Geräts exportieren

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Um alle aggregierten Bilder aller benutzerdefinierten Ordner eines HMI-Geräts zu exportieren, ist ein anderer Programmcode erforderlich.

Programmcode: Alle Bilder eines Geräts exportieren

Um die Bilder eines benutzerdefinierten Bilderordners eines HMI-Geräts und den Bildersystemordner zu exportieren, ändern Sie folgenden Programmcode:

```
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);
}
```

Programmcode: Alle Bilder eines benutzerdefinierten Ordners exportieren

Um die Bilder eines benutzerdefinierten Bilderordners eines HMI-Geräts und den Bildersystemordner zu exportieren, ändern Sie folgenden Programmcode:

```
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);
    }
}
```

Programmcode: Alle Bilder eines Geräts unabhängig vom Benutzer exportieren

Um alle Bilder zu exportieren, ändern Sie folgenden Programmcode:

```
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 Bild aus einem Bildordner exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten eines Bilds werden exportiert:

Bild	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name, Number, HelpText
Links öffnen	Template
Zusammensetzungen	• Layers • Animations Alle auf Runtime Advanced basierenden konfigurierten Animationen werden exportiert. • Events Alle auf Runtime Advanced basierenden konfigurierten Ereignisse werden exportiert. • Softkeys Alle konfigurierten Softkeys werden exportiert.

Für jede Schicht werden die folgenden Daten exportiert:

Hinweis

Standardmäßig besteht der Schichtname im TIA Portal aus leerem Text.

Wenn Sie den Schichtnamen im TIA Portal nicht ändern, ist der Name der exportierten Schicht leer. In diesem Fall ist der angezeigte Schichtname im TIA Portal von der Sprache der Benutzeroberfläche abhängig.

Wenn Sie den Schichtnamen im TIA Portal ändern, wird der geänderte Schichtname in allen entsprechenden Sprachen angezeigt.

Schicht	Daten
Attribute	Name, Index, VisibleES
Zusammensetzungen	ScreenItems (mit Bildelementen)

Nicht einbezogen werden in den Export:

- SCADA-spezifische Attribute
- Schichten, die keine Bildelemente enthalten und deren Attribute sich nicht von den Standardwerten unterscheiden.

Programmcode

Um ein einzelnes Bild aus dem Benutzerordner oder aus dem Systemordner eines HMI-Geräts zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a single screen from a screen folder
private static void ExportSingleScreenFromScreenFolder(HmiTarget hmitarget)
{
    ScreenUserFolder folder = hmitarget.ScreenFolder.Folders.Find("MyScreenFolder");
    //or ScreenSystemFolder folder = hmitarget.ScreenFolder;
    ScreenComposition screens = folder.Screens;
    Screen screen = screens.Find("Screen_1.xml");
    if (screen == null) return;
    {
        FileInfo info = new FileInfo(string.Format(@"D:\Samples\Screens\{0}\{1}.xml",
folder.Name, screen.Name));
        screen.Export(info, ExportOptions.WithDefaults);
    }
}
```

8.3.9.4 Bilder in ein HMI-Gerät importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die Bilder können nur in eine bestimmte Art von HMI-Gerät importiert werden. Das HMI-Gerät und das Gerät, aus dem die Bilder exportiert wurden, müssen den gleichen Gerätetyp besitzen.

Die folgenden Daten eines Bilds werden exportiert:

Bild	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name, Number, HelpText
Links öffnen	Templates
Zusammensetzungen	• Layers • Animations Alle für Bilder konfigurierbaren Animationen werden importiert. • Events Alle für Ereignisse konfigurierbaren Animationen werden importiert. • Softkeys Alle für Softkeys konfigurierbaren Animationen werden importiert.

Für jede Schicht werden die folgenden Daten importiert:

Hinweis

Wenn Sie vor dem Import für den Schichtnamen leeren Text angegeben haben, ist der angezeigte Schichtname im TIA Portal nach dem Import von der Sprache der Benutzeroberfläche abhängig.

Wenn Sie einen Schichtnamen zugewiesen haben, wird der angegebene Schichtname nach dem Import in allen entsprechenden Sprachen angezeigt.

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems

Einschränkungen

- Wenn die Breite und Höhe eines Bilds nicht den Abmessungen des Geräts entsprechen, wird der Importvorgang abgebrochen und eine Exception ausgelöst. Die Anpassung der enthaltenen Bildelemente wird nicht unterstützt. Aus diesem Grund können sich bestimmte Bildelemente außerhalb der Bildgrenzen befinden. In diesem Fall wird eine Compilerwarnung ausgegeben.
- Die Bildnummer muss für alle Bilder des Geräts jeweils eindeutig sein. Ein Bildimport wird abgebrochen, wenn ein Bild mit einer Bildnummer gefunden wird, die bereits im Gerät erstellt wurde. Wenn Sie noch keine Bildnummer zugewiesen haben, wird dem Bild während des Importvorgangs eine eindeutige Bildnummer zugeordnet.
- Die Anordnung der Bildelemente innerhalb der Z-Reihenfolge muss für jede Schicht im Bild eindeutig und lückenlos sein. Aus diesem Grund wird nach dem Import des Bilds eine Konsistenzprüfung durchgeführt, bei der die Anordnung gegebenenfalls repariert wird. Dieser Vorgang kann bei bestimmten Bildelementen zu geänderten „Tabindizes" führen. Sie können die Z-Reihenfolge der Bildelemente in der XML-Datei manuell ändern. Das Bildelement an erster Stelle befindet sich ganz am Ende der Z-Reihenfolge.

Hinweis

Sie können die Werte für Breite und Höhe eines Bildelements in der XML-Datei ändern, wenn für das Bildelement das Attribut „Größe an Inhalt anpassen“ aktiviert ist.

Hinweis

Import von Bildtypen aus der Bibliothek wird nicht unterstützt

Ab WinCC V12 SP1 können Sie in der Bibliothek ein Bild als Typ erstellen. Instanzen des im Projekt verwendeten Bildtyps können wie andere Bilder mit der TIA Portal Openness-Anwendung bearbeitet werden. Wenn Sie Bilder exportieren, werden die Instanzen von Bildtypen ohne die Typinformationen exportiert.

Wenn Sie diese Bilder wieder in das Projekt importieren, werden die Instanzen des Bildtyps überschrieben und die Instanz wird vom Bildtyp abgelöst.

Programmcode: Bilder in ein HMI-Gerät importieren

Um mit der For each-Schleife Bilder in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
//Imports all screens to an HMI device
private static void ImportScreensToHMITarget(HmiTarget hmitarget)
{
    FileInfo[] exportedScreens = new FileInfo[] {new FileInfo(@"D:\Samples\Import
\Screen_1.xml"), new FileInfo(@"D:\Samples\Import\Screen_2.xml")};
    ScreenUserFolder folder = hmitarget.ScreenFolder.Folders.Find("MyScreenFolder");
    foreach (FileInfo screenFileInfo in exportedScreens)
    {
        folder.Screens.Import(screenFileInfo, ImportOptions.Override);
    }
}
```

Programmcode: In einen neu erstellten Benutzerordner importieren

Um ein Bild in einen neu erstellten Benutzerordner eines HMI-Geräts zu importieren, ändern Sie folgenden Programmcode:

```
//Imports a single screen to a new created user folder of an HMI device
private static void ImportSingleScreenToNewFolderOfHMITarget(HmiTarget hmitarget)
{
    ScreenUserFolder folder = hmitarget.ScreenFolder.Folders.Create("MyFolder");
    folder.Screens.Import(new FileInfo(@"D:\Samples\Import\myScreens.xml"),
ImportOptions.Override);
}
```

8.3.9.5 Permanentfenster exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten des Permanentfensters werden exportiert:

Permanentfenster	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name
Zusammensetzungen	Layers

Für jede Schicht werden die folgenden Daten exportiert:

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems (mit Bildelementen)

Programmcode

Um ein Permanentfenster eines HMI-Geräts in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a permanent area
private static void ExportScreenoverview(HmiTarget hmitarget)
{
    ScreenOverview overview = hmitarget.ScreenOverview;
    if (overview == null) return;

    FileInfo info = new FileInfo(@"D:\Samples\Screens\ExportedOverview.xml");
    overview.Export(info, ExportOptions.WithDefaults);
}
```

8.3.9.6 Permanentfenster importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten des Permanentfensters werden importiert:

Permanentfenster	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name, Visible, Number
Zusammensetzungen	Layers

Für jede Schicht werden die folgenden Daten importiert:

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems (mit Bildelementen)

Wenn die Breite und Höhe eines Bilds nicht den Abmessungen des Geräts entsprechen, wird der Importvorgang abgebrochen und eine Exception ausgelöst. Die Anpassung der enthaltenen Geräteelemente (Bildelemente) wird nicht unterstützt. Aus diesem Grund können sich einige Geräteelemente außerhalb der Bildgrenzen befinden. In diesem Fall wird eine Compilerwarnung ausgegeben.

Die Anordnung der Geräteelemente im Permanentfenster muss eindeutig und lückenlos sein. Aus diesem Grund wird nach dem Import des Permanentfensters eine Konsistenzprüfung durchgeführt, bei der die Anordnung gegebenenfalls repariert wird. Dieser Vorgang kann bei bestimmten Geräteelementen zu geänderten „Tabindizes" führen.

Programmcode

Um ein Permanentfenster aus einer XML-Datei in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

```
//Imports a permanent area
private static void ImportScreenOverview(HmiTarget hmiTarget)
{
    FileInfo info = new FileInfo(@"D:\Samples\Screens\ExportedOverview.xml");
    hmiTarget.ImportScreenOverview(info, ImportOptions.Override);
}
```

8.3.9.7 Alle Bildvorlagen eines HMI-Geräts exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Pro Bildvorlage wird eine XML-Datei erstellt.

Da Sammelexporte nicht unterstützt werden, müssen Sie alle Bildvorlagen enumerieren und separat exportieren. Achten Sie dabei darauf, dass die verwendeten Bildvorlagennamen den Dateibenennungskonventionen Ihres Dateisystems entsprechen.

Programmcode: Alle Bildvorlagen eines Geräts exportieren

Um alle Bildvorlagen aus einem bestimmten Ordner zu exportieren, ändern Sie folgenden Programmcode:

```
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);
}
```

Programmcode: Alle Bildvorlagen eines bestimmten Ordners exportieren

Um alle Bildvorlagen zu exportieren, ändern Sie folgenden Programmcode:

```
//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 Bildvorlagen aus einem Ordner exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten der Bildvorlage werden exportiert:

Bildvorlagen	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name
Zusammensetzungen	• Layers • Animations Alle konfigurierten Animationen werden exportiert. SCADA-Animationen werden nicht exportiert. • Softkeys Alle konfigurierten Softkeys werden exportiert.

Für jede Schicht werden die folgenden Daten exportiert:

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems (mit Bildelementen)

Programmcode: Bildvorlage eines benutzerdefinierten Ordners exportieren

Um eine einzelne Bildvorlage aus dem Systemordner oder einem benutzerdefinierten Ordner zu exportieren, ändern Sie folgenden Programmcode:

```
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);
}
```

Programmcode: Alle Bildvorlagen eines benutzerdefinierten Ordners exportieren

Um alle Bildvorlagen aus einem bestimmten Ordner zu exportieren, ändern Sie folgenden Programmcode:

```
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);
    }
}
```

Programmcode: Alle Bildvorlagen eines bestimmten Ordners exportieren

Um alle Bildvorlagen zu exportieren, ändern Sie folgenden Programmcode:

```
//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 Bildvorlagen importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten einer Bildvorlage werden importiert:

Bildvorlage	Daten
Attribute	ActiveLayer, BackColor, Height, Width, Name, SetTabOrderInFront
Zusammensetzungen	• Layers • Animations Alle für Bilder konfigurierbaren Animationen werden importiert. • Softkeys Alle für Softkeys konfigurierbaren Animationen werden importiert.

Für jede Schicht werden die folgenden Daten importiert:

Schicht	Daten
Attribute	Name, Index
Zusammensetzungen	ScreenItems (mit Bildelementen)

Wenn die Breite und Höhe einer Bildvorlage nicht den Abmessungen des Geräts entsprechen, wird der Importvorgang abgebrochen und eine Exception ausgelöst. Die Anpassung der enthaltenen Bildelemente wird nicht unterstützt. Aus diesem Grund können sich bestimmte Bildelemente außerhalb der Bildgrenzen befinden. In diesem Fall wird eine Compilerwarnung ausgegeben.

Die Anordnung der Geräteelemente in der Bildvorlage muss eindeutig und lückenlos sein. Aus diesem Grund wird nach dem Import der Bildvorlage eine Konsistenzprüfung durchgeführt, bei der die Anordnung gegebenenfalls repariert wird. Dieser Vorgang kann bei bestimmten Bildelementen zu geänderten „Tabindizes" führen.

Programmcode: Allgemeiner Import

Um mit der For each-Schleife alle Bildvorlagen in ein HMI-Gerät zu importieren, ändern Sie folgenden Programmcode:

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

Programmcode: In einen neu erstellten Benutzerordner importieren

Um eine Bildvorlage in einen neu erstellten Benutzerordner eines HMI-Geräts zu importieren, ändern Sie folgenden Programmcode:

```
//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 Exportieren eines Pop-up-Bilds

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten des Pop-Up-Bilds werden exportiert:

Bildvorlagen	Daten
Attribute	ActiveLayer, BackColor, GridColor, Height, Name, ScrollbarBackgroundColor, ScrollbarForegroundColor, Width
Zusammensetzungen	• Layers • Events Alle konfigurierten Ereignisse werden exportiert.

Für jede Schicht werden die folgenden Daten exportiert:

Schicht	Daten
Attribute	Name, Index, VisibleES
Zusammensetzungen	ScreenItems Alle exportierbaren Bildobjekte werden exportiert.

Programmcode: Exportieren eines Pop-up-Bilds aus einem Ordner

Um ein einzelnes Pop-up-Bild aus dem Systemordner oder einem benutzerdefinierten Ordner zu exportieren, ändern Sie folgenden Programmcode:

```
//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 Importieren eines Pop-up-Bildes

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten eines Pop-up-Bilds werden importiert:

Bildvorlagen	Daten
Attribute	ActiveLayer, BackColor, GridColor, Height, Name, ScrollbarBackgroundColor, ScrollbarForegroundColor, Width
Zusammensetzungen	• Layers • Events Alle konfigurierten Ereignisse werden exportiert.

Die Existenz folgender Attribute ist für das Importieren verpflichtend:

- Name
- Height
- Width

Für jede Schicht werden die folgenden Daten importiert:

Schicht	Daten
Attribute	Name, Index, VisibleES
Zusammensetzungen	ScreenItems Alle importierbaren Bildobjekte werden importiert.

Einschränkungen

Wenn das Gerät keine Pop-up-Bilder unterstützt, wird der Kopiervorgang abgebrochen und eine Ausnahme ausgelöst.

Wenn die Breite und Höhe eines Pop-up-Bilds nicht den Abmessungen des Geräts entsprechen, wird der Importvorgang abgebrochen und eine Exception ausgelöst.

- Minimale Höhe = 1 Pixel
- Minimale Breite = 1 Pixel
- Max. Höhe = sechsfache Höhe des Gerätebildschirms
- Max. Breite = zweifache Breite des Gerätebildschirms
- Bei Geräten mit Runtime-Version V13 SP1 entspricht die maximale Höhe und die maximale Breite der Höhe und Breite des Gerätebildschirms.

Programmcode: Importieren eines Pop-up-Bilds in einen Ordner

Um ein Pop-up-Bild in einen Pop-up-Bild-Systemordner oder in einen benutzerdefinierten Ordner zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports a pop-up screen to an HMI device
private static void ImportPopupScreenToHMITarget(HmiTarget hmitarget)
{
    FileInfo info = new FileInfo(string.Format(@"D:\Samples\Screens\PopupScreen.xml"));
    hmitarget.ScreenPopupFolder.ScreenPopups.Import(info, ImportOptions.None);
}
```

8.3.9.12 Exportieren eines Slide-in-Bilds

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten und Werte des Slide-in-Bilds werden exportiert:

Bildvorlagen	Daten	
Attribute	Activate	false
	ActiveLayer	0
	BackColor	(182; 182; 182)
	GridColor	(0; 0; 0)
	Dimension	427 Das Attribut "Dimension" gibt entweder die Breite oder die Höhe des Slide-in-Bilds an, abhängig von der Art des Slide-in-Bilds.
	LineColor1	(223; 223; 223)
	LineColor2	(32; 32; 32)
	OperatableAreaColor	(128; 128; 128)
	SlideinType	Oben, Unten, Links, Rechts Slide-in-Bilder haben keinen Namen, sondern einen SlideinType.
	Visibility	FadeOut
Zusammensetzungen	Layers	

Hinweis

Slide-in-Bilder haben keinen Namen, sondern einen SlideinType.

Für jede Schicht werden die folgenden Daten exportiert:

Schicht	Daten	
Attribute	Name,	
	Index	
	VisibleES	
Zusammensetzungen	ScreenItems	Alle exportierbaren Bildobjekte werden exportiert.

Programmcode: Exportieren eines Slide-in-Bilds

Um ein einzelnes Slide-in-Bild aus dem Systemordner zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a single slide-in screen
private static void ExportSingleSlideinScreen(HmiTarget hmitarget)
{
    ScreenSlideinSystemFolder systemFolder = hmitarget.ScreenSlideinFolder;
    var screens = systemFolder.ScreenSlideins;
    ScreenSlidein slidein = screens.Find(SlideinType.Bottom);
    if (slidein == null) return;

    FileInfo info = new FileInfo(string.Format(@"D:\Samples\Screens\{0}\{1}.xml"));
    slidein.Export(info, ExportOptions.WithDefaults);
}
```

8.3.9.13 Importieren eines Slide-in-Bilds

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten und Werte eines Slide-in-Bilds werden importiert:

Bildvorlagen	Daten
Attribute	Activate = false ActiveLayer = 0 Authorization BackColor = (182; 182; 182) Dimension = 427 Das Attribut „Dimension“ gibt entweder die Breite oder die Höhe des Slide-in-Bilds an, abhängig davon, welches der zwei Attribute für die Art des Slide-in-Bilds geändert werden kann. GridColor = (0; 0; 0) LineColor1 = (223; 223; 223) LineColor2 = (32; 32; 32) OperateableAreaColor = (128; 128; 128) SlideinType = Top, Bottom, Left, Right Visibility = FadeOut
Zusammensetzungen	Layers

Die Existenz des folgenden Attributs ist für das Importieren verpflichtend:

- SlideinType

Für jede Schicht werden die folgenden Daten importiert:

Schicht	Daten
Attribute	Name, Index, VisibleES
Zusammensetzungen	ScreenItems Alle importierbaren Bildobjekte werden importiert.

Einschränkungen

- Wenn das Gerät keine Slide-in-Bilder unterstützt, wird der Import abgebrochen und eine Ausnahme ausgelöst.
- Wird ein Slide-in-Bild von einem anderen Element referenziert, ist das Slide-in-Bild über openlink zu referenzieren und nicht über SlideinType, z. B. in der Systemfunktion „ShowSlideinScreen“).
  Die folgende Tabelle zeigt die Abbildung des Attributs "SlideinType" mit dem entsprechenden openlink:

SlideinType	Openlink-Name
Top	GraphX_Slidein_Top
Right	GraphX_Slidein_Right
Bottom	GraphX_Slidein_Bottom
Left	GraphX_Slidein_Left

Programmcode: Importieren eines Slide-in-Bilds in einen Ordner

Um ein Slide-in-Bild in einen Slide-in-Bild-Systemordner zu importieren, ändern Sie folgenden Programmcode:

```
//Imports a slide-in screen to an HMI device
private static void ImportSlideinScreenToHMITarget(HmiTarget hmitarget)
{
    FileInfo info = new FileInfo(@"D:\Samples\Screens\SlideInScreen.xml");
    hmitarget.ScreenSlideinFolder.ScreenSlideins.Import(info, ImportOptions.None);
}
```

8.3.9.14 Bild mit Eingabemaske exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten einer Bildbausteininstanz in einem Bild werden exportiert:

Bild	Daten
Attribute	Left, Top, Width, Height, ObjectName, Resizing, TabIndex, FaceplateTypeName
Schnittstellenattribute	Alle konfigurierten Schnittstellenattribute einer Bildbausteininstanz werden als exportierbare Bildelemente exportiert.
Zusammensetzungen	• Animationen Alle bewegten Animationen werden exportiert. Variablenanimationen basieren auf den Attributen der Schnittstelle. • Ereignisse Alle konfigurierten Ereignisse werden exportiert.

Beachten Sie die folgenden Vorgaben für exportierte Attribute von Bildbausteininstanzen:

- Resizing
  Das Attribut „Resizing“ wird in jedem Fall exportiert, unabhängig von den Exportoptionen.
- FaceplateTypeName
  Das Attribut "FaceplateTypeName" identifiziert den entsprechenden Typ und die Version des Bildbausteins, z. B. „Faceplate_1 V 0.0.2“.

Hinweis

Bildbausteintyp in einem Bibliotheksordner

Befindet sich ein Bildbausteintyp in einem Bibliotheksordner, benötigen Sie den gesamten Pfad und Namen zur Identifizierung des Bildbausteintyps. Das Schlüsselwort „@$@“ wird verwendet, um Ordner und/oder Namen von Bildbausteintypen zu trennen, z. B. „Folder_1@$@SubFolder_1@$@Faceplate_1 V 0.0.2“.

Die folgenden Daten von Bildelementen innerhalb einer Bildbausteininstanz sind vom Export ausgeschlossen:

Bildelement	Attribut
E/A-Feld	Flashing on limit violation
Grafisches E/A-Feld	Fit embedded graphic object to screen size

Programmcode

Um ein einzelnes Bild mit einer Bildbausteininstanz zu exportieren, ändern Sie den folgenden Programmcode:

```
//Exports a single screen including a faceplate instance
private static void ExportSingleScreenWithFaceplateInstance(HmiTarget hmitarget)
{
    ScreenFolder folder = hmitarget.ScreenFolder.Folders.Find("MyScreenFolder");
    ScreenComposition screens = folder.Screens;
    Screen screen = screens.Find("ScreenWithFaceplateName");
    if (screen == null) return;
    {
       FileInfo info = new FileInfo(string.Format(@"D:\Samples\Faceplates\{0}\{1}.xml",
folder.Name, screen.Name));
       screen.Export(info, ExportOptions.WithDefaults);
    }
}
```

8.3.9.15 Bild mit Eingabemaske importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Die folgenden Daten einer Bildbausteininstanz in einem Bild werden importiert:

Bild	Daten
Attribute	Left, Top, Width, Height, ObjectName, Resizing, TabIndex, FaceplateTypeName
Schnittstellenattribute	Alle konfigurierten Schnittstellenattribute einer Bildbausteininstanz werden als importierbare Bildelemente importiert.
Zusammensetzungen	• Animationen Alle bewegten Animationen werden importiert. Variablenanimationen basieren auf den Attributen der Schnittstelle. • Ereignisse Alle konfigurierten Ereignisse werden importiert.

Die Existenz folgender Attribute ist für das Importieren verpflichtend:

- ObjectName
- FaceplateTypeName

Die folgenden Daten von Bildelementen innerhalb einer Bildbausteininstanz sind von Export und Import ausgeschlossen:

Bildelement	Attribut
E/A-Feld	Flashing on limit violation
Grafisches E/A-Feld	Fit embedded graphic object to screen size

Einschränkungen

- Bildbaustein, Ereignis oder Schnittstellenattribut unbekannt
  Wenn ein faceplate type name, ein event name oder ein interface attribute name in der Importdatei angegeben ist, der nicht im Projekt existiert, wird der Import abgebrochen und eine Ausnahme ausgegeben.
- Skalierungsverhalten einer Bildbausteininstanz
  Das Attribut „Resizing“ wird in jedem Fall importiert, unabhängig von den Exportoptionen.
  Beispiele:
  Ist "Resizing" auf "KeepRatio" gesetzt, dann wird das Attribut "Height" verwendet, um den Attributwert "Width" zu berechnen.
  - Die Größe eines Bildbausteintyps beträgt 100 x 100 Pixel. Wenn eine Bildbausteininstanz mit einer Größe von 300 x 100 Pixel importiert wird und der Wert "FixedSize" auf das Attribut "Resizing" eingestellt ist, ist der Import erfolgreich und die Bildbausteingröße wird auf 100 x 100 Pixel gesetzt.
  - Die Größe eines Bildbausteintyps beträgt 100 x 50 Pixel. Wenn eine Bildbausteininstanz mit einer Größe von 100 x 100 Pixel importiert wird und der Wert "KeepRatio" auf das Attribut "Resizing" eingestellt ist, ist der Import erfolgreich und die Bildbausteingröße wird auf 200 x 100 Pixel gesetzt.

  Hinweis

  Skalierungsverhalten von importierten Bildbausteininstanzen

  Die Werte von „Resizing“ und die Werte der Schnittstellenattribute können die Größe der importierten Bildbausteininstanz und sogar die Größe der darin enthaltenen Bildelemente beeinflussen.

  Um ungewünschte Änderungen am Aussehen einer Bildbausteininstanz zu vermeiden, importieren Sie einen Bildbaustein in der ursprünglichen Größe oder ohne die Attributwerte "Width" und "Height".

- Abweichende Schnittstellenattributwerte
  - Wenn Sie die Attribute für den Import ändern, wird der letzte verwendete Schnittstellenattributwert importiert.
  - Wenn die Attribute voneinander abhängig sind, können andere Attribute während des Imports geändert werden.
    Beispiel: Ein Bildbaustein enthält ein E/A-Feld. Das Attribut „Betriebsart“ wird mit einem Schnittstellenattribut verbunden. Wenn Sie erst die Betriebsart auf „Ausgang“ und dann das Attribut „Versteckter Eingang“ auf wahr setzen, wird der Wert von „Versteckter Eingang“ nach dem Import nicht verwendet. Das Attribut „Versteckter Eingang“ wurde bei der ersten Änderung auf schreibgeschützt gesetzt und der Wert kann nicht verwendet werden.
  - Wenn ein Attributwert die Einschränkungen von WinCC nicht erfüllt, wird der Wert des Bildbausteintyps angezeigt.
    Beispiel: Der Anzeigebereich eines Zeigerinstrument wird auf 10 - 80 eingestellt. Die Attribute „Maximalwert“ und „Minimalwert“ werden als Schnittstellenattribute konfiguriert. Wenn Sie einen Minimalwert einstellen, der den Maximalwert übersteigt, z. B. 100, dann wird der Wert des Bildbausteintyps für „Minimalwert“ nach dem Import angezeigt.

– Wenn ein Schnittstellenattribut mit mehreren Attributen von Bildelementen innerhalb des Bildbausteintyps verbunden ist, dann zeigt das Schnittstellenattribut an der Bildbausteininstanz den verwendeten Attributwert des ersten verbundenen Bildelements an.
  Beispiel: Ein Bildbaustein enthält zwei Zeigerinstrumente mit abweichenden Maximalwerten. Die Minimalwerte beider Zeigerinstrumente werden mit einem einzelnen Schnittstellenattribut verbunden.
  Wenn Sie erst einen Minimalwert einstellen, der für beide Zeigerinstrumente anwendbar ist, werden beide Werte eingestellt.
  Wenn Sie dann einen Wert einstellen, der nur für das zweite Zeigerinstrument anwendbar ist, wird der Wert nur für das zweite Zeigerinstrument eingestellt, der Wert des ersten Zeigerinstruments wird hingegen als Schnittstellenattribut angezeigt.

Programmcode: Bilder mit einer Bildbausteininstanz importieren

Um ein Bild mit einer Bildbausteininstanz zu importieren, ändern Sie den folgenden Programmcode:

```
//Imports single screen including a faceplate instance
private static void ImportSingleScreenWithFaceplateInstance(HmiTarget hmitarget)
{
    FileInfo info = new FileInfo(@"D:\Samples\Screens\ScreenFaceplate.xml");
    hmitarget.ScreenFolder.Screens.Import(info, ImportOptions.None);
}
```

8.4 Import/Export von Daten eines PLC-Geräts

8.4.1 Bausteine

8.4.1.1 XML-Struktur des Abschnitts Bausteinschnittstelle

Grundprinzip

Die Daten in der Exportdatei vom Import/Export sind mit Verweis auf eine Grundstruktur gegliedert. Jede Importdatei muss die grundlegenden strukturellen Bedingungen erfüllen.

Die Exportdatei enthält alle bearbeiteten Variablen und Konstanten des Schnittstellenabschnitts eines exportierten Bausteins. Alle Attribute mit "`ReadOnly` = "TRUE"" und "`Informative` = "TRUE"" werden ausgeschlossen.

Wenn die Information redundant ist, muss sie in der Import-XML-Datei und der Projektdatei identisch sein. Andernfalls wird beim Import eine wiederherstellbare Ausnahme ausgelöst.

Die Projektdaten enthalten mehr Daten als dei Import-XML-Datei, z. B. kann ein externer Typ zusätzliche Mitglieder haben.

Nur schreibbare Werte können über TIA Portal Openness XML importiert werden.

Abhängig von den Exporteinstellungen in TIA Portal Openness enthält die Exportdatei einen festgelegten Satz von Attributen und Elementen. Die aus höheren Versionen des Produkts exportierte XML ist beim Importvorgang in niedrigeren Versionen des TIA Portals nicht kompatibel.

Grundstruktur

Der Schnittstellenabschnitt eines exportierten Bausteins ist im Element <`Interface`> in der SimaticML eines Bausteins enthalten. Das Stammobjekt ist das Element `<Sections>`, das den Schnittstellenabschnitt eines exportierten Bausteins darstellt. Die Reihenfolge der folgenden Beschreibung von Elementen stellt die erforderliche Reihenfolge in der Eingabedatei dar.

```
<Interface>
    <Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v1">
        <Section Name="Input">
            <Member Name="input1" Datatype="Bool" Remanence="Volatile" Accessibility="Public">
              <AttributeList>
                  ...
              </AttributeList>
            </Member>
            <Member Name="input2" Datatype="Bool" Remanence="Volatile" Accessibility="Public">
                <AttributeList>
                  ...
                </AttributeList>
            </Member>
        </Section>
        <Section Name="Output">
            <Member Name="output1" Datatype="Bool" Remanence="Volatile" Accessibility="Public">
                <AttributeList>
                    ...
                </AttributeList>
            </Member>
        </Section>
        <Section Name="InOut" />
        <Section Name="Static" />
        <Section Name="Temp" />
        <Section Name="Constant" />
    </Sections>
</Interface>
```

- Abschnitt
  Abschnitt stellt einen einzelnen Parameter oder Lokaldaten eines Programmbausteins dar.
- Mitglied
  Mitglied stellt die im Programmbaustein verwendeten Variablen oder Konstanten dar. Abhängig vom Datentyp einer Variablen können Mitglieder verschachtelt sein oder weitere strukturelle Unterelemente haben.
  Beim Datentyp "ARRAY" stellt das strukturelle Element "`Subelement Path`" z. B. den Index von Komponenten eines Array-Elements dar.
  Es werden nur die Mitglieder exportiert, die vom Anwender bearbeitet wurden.
- AttributeList
  Die `<AttributeList>` umfasst alle definierten Attribute eines Mitglieds. Attribute, die vom System festgelegt oder von einem Standardwert zugewiesen sind, werden in der XML-Struktur nicht aufgeführt.
  Die Mitgliedsattribute `<ReadOnly>` und `<Informative>` werden nur in die XML-Exportdatei geschrieben, wenn ihr Wert TRUE ist.

- StartValue
  Das Element <StartValue> wird nur geschrieben, wenn der Standardwert der Variablen oder Konstanten vom Anwender festgelegt ist.

```
...
<Member Name="Static_1" Datatype="Int">
 ...
 <StartValue>1</StartValue>
</Member>
...
```

- Kommentar
  Das Element <Comment> wird geschrieben, wenn es vom Anwender festgelegt ist. Kommentare einer Variablen oder Konstanten werden als mehrsprachiger Text exportiert:

```
...
 <Member Name="Static_3" Datatype="Struct">
   <AttributeList>
     <BooleanAttribute Name="ExternalAccessible" SystemDefined="true">false</BooleanAttribute>
     <BooleanAttribute Name="ExternalVisible" SystemDefined="true">false</BooleanAttribute>
     <BooleanAttribute Name="ExternalWritable" SystemDefined="true">false</BooleanAttribute>
   </AttributeList>
   <Comment>
     <MultiLanguageText Lang="de-DE">An individual comment</MultiLanguageText>
   </Comment>
 </Member>
...
```

Attribute

- Hauptattribute
  Die Hauptattribute werden in das Element `<Member>` in der XML-Struktur geschrieben.

```
...
<Member Name="Static_1" Datatype="&amp;quot;User_data_type_1&amp;quot;" Remanence="Retain">
  ...
</Member>
...
```

Die folgende Tabelle zeigt die Hauptattribute einer Variablen oder Konstanten am Abschnitt Bausteinschnittstelle.

Name	Datentyp	Standardeinstellung	Importbedingung	Kommentar
Name	STRING	-	Erforderlich	
Datentyp	ENUM	-	Erforderlich	
Version	STRING	-	Optional	
Remanenz	ENUM	NonRetain	-	Wird nur geschrieben, wenn es sich nicht um die Standardeinstellung handelt
Zugänglichkeit	ENUM	Öffentlich	-	Vom System vordefiniert Kann vom Anwender nicht geändert werden
Informativ	BOOL	FALSE	-	

Mitglieder mit dem Merker "`Informative`" werden beim Import ignoriert. Wenn das Attribut gelöscht oder auf FALSE gesetzt ist, wird eine Ausnahme ausgelöst.

Hinweis

Remanenzeinstellungen "Im IDB setzen"

Wenn der Remanenzwert einer Variablen oder Konstanten "Im IDB setzen" ist, muss die im IDB festgelegte Remanenz für alle anderen Variablen und Konstanten mit dem Remanenzwert "SetInIDB" die gleiche sein.

Das erste importierte Mitglied mit dem Attribut "Im IDB setzen" legt die erwartete Remanenz im IDB für die folgenden Variablen und Konstanten mit dem Remanenzwert "SetInIDB" fest.

- Systemdefinierte Mitgliedsattribute
  Systemdefinierte Mitgliedsattribute werden im Element `<AttributeList>` aufgelistet. Systemdefinierte Mitgliedsattribute werden mit dem Merker `<Informative>` gekennzeichnet und beim Import ignoriert.

```
...
 <Member Name="Static_3" Datatype="Struct">
   <AttributeList>
     <BooleanAttribute Name="ExternalAccessible" SystemDefined="true">false</BooleanAttribute>
     <BooleanAttribute Name="ExternalVisible" SystemDefined="true">false</BooleanAttribute>
     <BooleanAttribute Name="ExternalWritable" SystemDefined="true">false</BooleanAttribute>
   </AttributeList>
   <Comment>
     <MultiLanguageText Lang="de-DE">An individual comment</MultiLanguageText>
   </Comment>
 </Member>
...
```

Name	Typ	Standardeinstellung	SimaticML schreibgeschützt (informativ)	Kommentar
At	String	""	FALSE	Mitgliedsanteile sind mit einem anderen Mitglied in dieser Struktur versetzt
SetPoint	Bool	FALSE	FALSE	Mitglied kann mit Arbeitsspeicher synchronisiert werden
UserReadOnly	Bool	FALSE	TRUE	Anwender kann keine Mitgliedsattribute ändern (einschl. Name)
UserDeletable	Bool	TRUE	TRUE	Mitglied kann im Editor nicht gelöscht werden
HmiAccessible	Bool	TRUE	FALSE	Kein HMI-Zugriff, kein Strukturelement
HmiVisible	Bool	TRUE	FALSE	Filter zum Verringern der Anzahl Mitglieder, die zuerst angezeigt werden
Offset	Int	-	TRUE	DB, FB, FC (Temp). Bei klassischen PLCs und Plus PLCs, wo die klassische Remanenz eingestellt ist.

Name	Typ	Standardeinstellung	SimaticML schreibgeschützt (informativ)	Kommentar
PaddedSize	Int	-	TRUE	DB, FB, FC (Temp). Bei klassischen PLCs und Plus PLCs, wo die klassische Remanenz eingestellt ist. Nur bei Arrays.
HiddenAssignment	Bool	FALSE	FALSE	Zuweisung bei Aufruf ausblenden, wenn sie PredefinedAssignment entspricht.
PredefinedAssingment	String	""	FALSE	Eingabe für den verwendeten Parameter, wenn der Aufruf platziert wird.
ReadOnlyAssignment	Bool	FALSE	FALSE	Der Anwender kann die vordefinierte Zuweisung beim Aufruf nicht ändern.
UserVisible	Bool	TRUE	TRUE	Dieses Mitglied wird in der UI nicht angezeigt.
HmiReadOnly	Bool	TRUE	TRUE	Dieses Mitglied für die HMI schreibgeschützt.
CodeReadOnly	Bool	FALSE	TRUE	-

- Anwenderdefinierte Attribute
  Anwenderdefinierte Attribute werden mit dem Merker <ReadOnly> gekennzeichnet. Mitglieder mit diesem Merker werden beim Import ignoriert. Wenn der Merker gelöscht oder auf FALSE gesetzt ist, wird eine Ausnahme ausgelöst.
  Nicht bearbeitete anwenderdefinierte Attribute werden vom Export ausgeschlossen.

Name	Typ	Standardeinstellung	SimaticML schreibgeschützt (informativ)	Kommentar
CFC	IBlockAttribute	---	FALSE	Dies ist eine Ladung

Datentyp "STRUCT"

Die Komponenten des Datentyps "STRUCT" werden in der XML-Struktur einer Import-/Exportdatei als verschachtelte Mitglieder dargestellt:

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype="Struct">
      <!-- Basic struct -->
      <Member Name="Static_1" Datatype="Int">
        <!-- First Member of struct -->
        <StartValue>1</StartValue>
      </Member>
      <Member Name="Static_2" Datatype="Int">
        <!-- Second Member of struct -->
      </Member>
      <Member Name="Static_3" Datatype="Struct">
        <!-- A subsequent struct -->
        <Member Name="Static_1" Datatype="Int">
          <!-- First Member of the subsequent struct -->
          <StartValue>3</StartValue>
        </Member>
        <Member Name="Static_2" Datatype="Int">
          <!-- Second Member of the subsequent struct -->
        </Member>
      </Member>
    </Member>
  </Section>
</Sections>
```

Datentyp "ARRAY" Basistyp

Die Komponenten des Basisdatentyps "ARRAY" werden in der XML-Struktur einer Import-/Exportdatei als Unterelemente mit dem Attribut "`Path`" dargestellt:

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype="Array[0..2] of Int" Remanence="Retain">
      <!-- Basic Array -->
      <Subelement Path="0">
        <!-- First Array Component-->
        <StartValue>1</StartValue>
      </Subelement>
      <Subelement Path="1">
        <!-- Second Array Component-->
        <StartValue>2</StartValue>
      </Subelement>
    </Member>
  </Section>
</Sections>
```

Datentyp "ARRAY" von UDT

Die Komponenten des Datentyps "ARRAY" eines UDT werden in der XML-Struktur einer Import-/Exportdatei als neues Element `<sections>` in einem Element `<member>` dargestellt. Die Mitglieder im neuen Abschnitt für UDT in einem ARRAY sind als Unterelemente mit dem Attribut `"Path"` zugewiesen:

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype="&amp;quot;User_data_type_1&amp;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 &amp;quot;User_data_type_1&amp;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>
```

Datentyp "ARRAY" in "ARRAY"

Die Komponenten des Datentyps "ARRAY" in einem anderen ARRAY werden in der XML-Struktur einer Import-/Exportdatei als Unterelemente mit dem Attribut `"Path"` dargestellt.

Die Mitglieder in einem anderen ARRAY werden als Unterelemente mit dem Attribut`"Path"` zugewiesen, wenn die Komponente vom Anwender bearbeitet wird:

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Static">
    <Member Name="Static_1" Datatype="Array[0..2] of Struct">
      <Member Name="Static_1" Datatype="Int" />
      <Member Name="Static_2" Datatype="Array[0..1, 0..3, -9..-2] of Struct">
        <Member Name="Static_1" Datatype="Int">
          <Subelement Path="0,0,3,-5">
            <StartValue>1</StartValue>
          </Subelement>
        </Member>
        <Subelement Path="0,0,2,-6">
          <Comment>
            <MultiLanguageText Lang="de-DE">A individual comment</MultiLanguageText>
          </Comment>
        </Subelement>
      </Member>
    </Member>
  </Section>
</Sections>
```

PLC-Datentypen (UDT)

Die XML-Struktur eines PLC-Datentyps ist von den Exporteinstellungen in TIA Portal Openness abhängig.

- `ExportOptions.None`
  Mitglieder des PLC-Datentyps werden nur geschrieben, wenn der Standardwert von mindestens einer der Komponenten vom Anwender festgelegt ist. Bei diesen Mitgliedern werden nur die zwei zusätzlichen Attribute "`Name`" und "`Datatype`" geschrieben, um das Mitglied zu identifizieren, zu dem der `<StartValue>` gehört. Andere Mitglieder und Attribute werden nicht geschrieben.
- `ExportOptions.WithDefaults`
  Die folgenden Attribute werden immer geschrieben:
  - `Name`
  - `Datatype`
  - `ExternalAccessible`
  - `ExternalVisible`
  - `ExternalWritable`
  - `SetPoint`
  - `StartValue`
    Sie werden nur in XML geschrieben, wenn der Standardwert in diesem Typ vom Anwender festgelegt ist. Wenn er nur im PLC-Datentyp festgelegt ist, wird er nicht geschrieben.
- `ExportOptions.ReadOnly`
  Bei PLC-Datentypen führt diese Einstellung nicht zu einem aussagekräftigen Ergebnis. In Kombination mit anderen Einstellungen hat dies keine Auswirkung auf das Ergebnis.

Überlagerte Variablen

Wenn eine Variable mit einem neuen Datentyp überlagert wird, werden die Mitglieder in der XML-Struktur des neuen Datentyps dargestellt. Die folgende XML-Struktur zeigt einen Datentyp WORD, der mit einem ARRAY von BYTE überlagert ist.

```
<Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
  <Section Name="Input" />
  <Section Name="Output" />
  <Section Name="InOut" />
  <Section Name="Static">
    <Member Name="Static_1" Datatype="Word">
      <!-- Basic Member -->
      <StartValue>16#17</StartValue>
    </Member>
    <Member Name="Static_2" Datatype="Array[0..1] of Byte">
      <AttributeList>
        <StringAttribute Name="At" SystemDefined="true">Static_1</StringAttribute>
      </AttributeList>
        <!-- The AT member -->
      <Subelement Path="0">
        <!-- First overlay byte -->
      </Subelement>
      <Subelement Path="1">
        <!-- Second overlay byte -->
      </Subelement>
    </Member>
  </Section>
  <Section Name="Temp" />
  <Section Name="Constant" />
</Sections>
```

Bausteinschnittstelle

Alle Attribute mit `ReadOnly` = "TRUE" und `Informative` = "FALSE" werden ausgeschlossen. Die XML-Struktur einer Bausteinschnittstelle ist von den Exporteinstellungen in TIA Portal Openness abhängig.

- `ExportOptions.None`
  Diese Einstellung exportiert nur die geänderten Daten oder die Daten, die von den Standardwerten abweichen.
  Wenn die Attributdefinition keinen Standardwert angibt, wird das Attribut immer geschrieben.
  Die Exportdatei enthält auch alle Werte, die für den anschließenden Datenimport obligatorisch sind.
- `ExportOptions.WithDefaults`
  Die folgenden Attribute werden immer geschrieben:
  - `Name`
  - `Datatype`
  - `HmiAccessible` exportiert als `ExternalAccessible`
  - `HmiVisible` exportiert als `ExternalVisible`
  - `ExternalWritable`
  - `SetPoint` (sofern vorhanden)
  - `Offset` (sofern vorhanden)
  - `PaddedSize` (sofern vorhanden)

  Alle anderen Attribute werden nur geschrieben, wenn sich ihre Werte vom Standardwert unterscheiden.
  Das Element `<StartValue>` wird nur in XML geschrieben, wenn es explizit festgelegt wurde.
- `ExportOptions.ReadOnly`
  Bei Bausteinschnittstellen führt diese Einstellung nicht zu einem aussagekräftigen Ergebnis. In Kombination mit anderen Einstellungen hat dies keine Auswirkung auf das Ergebnis.

8.4.1.2 Änderungen des Objektmodells und Dateiformat XML

Einleitung

Um eine kundenspezifisch erzeugte oder editierte XML-Datei über TIA Portal Openness erfolgreich in das TIA Portal zu importieren, muss die Datei definierten Schemas entsprechen.

Die XML-Dateien bestehen immer aus zwei Hauptteilen:

- Schnittstelle
- Übersetzungseinheit

Die Schemas, denen die Dateien entsprechen müssen, werden nachstehend erläutert.

Schnittstelle

Eine Schnittstelle kann mehrere Abschnitte enthalten (z. B. Input, InOut, Static): Sie finden alle diese Abschnitte im folgenden Verzeichnis:

C:\Program Files\Siemens\Automation\Portal V14\PublicAPI\V14 SP1\Schemas \SW.InterfaceSections_v2.xsd

Übersetzungseinheit

Es gibt separate Schemas für die Übersetzungseinheiten der Bausteine GRAPH, KOP/FUP und AWL. Sie finden diese Schemas in den folgenden Verzeichnissen:

- GRAPH: C:\Program Files\Siemens\Automation\Portal V ...\PublicAPI\Schemas \SW.PlcBlocks.Graph.xsd
- KOP/FUP: C:\Program Files\Siemens\Automation\Portal V ...\PublicAPI\Schemas \SW.PlcBlocks.LADFBD.xsd
- AWL: C:\Program Files\Siemens\Automation\Portal V ...\PublicAPI\Schemas \SW.PlcBlocks.STL.xsd

Subschemas

Es gibt folgende zusätzliche Schemadefinitionen, die von allen Übersetzungseinheiten verwendet werden:

- Zugriff
- Allgemein

Zugriff

Der Zugriffsknoten beschreibt zum Beispiel:

- lokale/globale Mitglieder und konstante Nutzungen
- FB-, FC-, Anweisungsaufrufe
- DBs für Aufrufe

Sie finden das Zugriffsschema im folgenden Verzeichnis:

C:\Program Files\Siemens\Automation\Portal V14\PublicAPI\V ...\Schemas \SW.PlcBlocks.Access.xsd

Allgemein

Hierunter fallen die allgemein verwendeten Attribute und Elemente, zum Beispiel Kommentare verschiedener Art, Texte und Token.

Sie finden das allgemeine Schema im folgenden Verzeichnis:

C:\Program Files\Siemens\Automation\Portal V14\PublicAPI\V ...\Schemas\SW.Common.xsd

8.4.1.3 Bausteine exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)
- Die PLC ist nicht online.

Verwendung

Die API-Schnittstelle unterstützt das Exportieren konsistenter Bausteine und Anwenderdatentypen in eine XML-Datei.

Die XML-Datei erhält den Namen des Bausteins. Die folgenden Bausteintypen werden unterstützt:

- Funktionsbausteine (FB)
- Funktionen (FC)
- Organisationsbausteine (OB)
- Globale Datenbausteine (DB)

Die folgenden Programmiersprachen werden unterstützt:

- AWL
- FUP
- KOP
- GRAPH
- SCL

Für alle Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions` in alle Bausteine exportiert (siehe Export von Projektierungsdaten (Seite 427)). In Fettdruck dargestellten Attribute werden immer exportiert.

Weitere Informationen finden Sie im Informationssystem des TIA Portals unter „Übersicht über die Bausteinattribute“.

Attribut	**Typ**	**Standardwert**	**Schreibgeschützt**
AutoNumber	Bool	true	false
CodeModifiedDate	DateTime	-	true
CompileDate	DateTime	-	true
CreationDate	DateTime	-	true
HeaderAuthor	String	""	false

Attribut	Typ	Standardwert	Schreibgeschützt
HeaderFamily	String	""	false
HeaderName	String	""	false
HeaderVersion	String	„0,1“	false
Interface	String	leere Schnittstelle	false
InterfaceModifiedDate	DateTime	-	true
IsConsistent	Bool	-	true
IsKnowHowProtected[1]	Bool	false	true
IsWriteProtected	Bool	false	true
MemoryLayout	enum MemoryLayout	-	false
ModifiedDate	DateTime	-	true
Name	String	-	false
Number	Int32	nächste verfügbare Nummer	false
ParameterModified	DateTime	-	true
PLCSimAdvancedSupport	Bool	false	true
ProgrammingLanguage	enum ProgrammingLanguage	-	false
StructureModified	DateTime	-	true

[1] Das Attribut IsKnowHowProtected gilt auch für UDT.

Für ArrayDB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für ArrayDB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
ArrayDataType	String	-	true
ArrayLimitUpperBound	Int32	-	true

Für DB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für DB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
IsOnlyStoredInLoadMemory	Bool	false	false
IsPLCDB	Bool	false	false
IsWriteProtectedInAS	Bool	false	false

Für FB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für FB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
AssignedProDiagFB	String	-	-
ISMultiInstanceCapable	Bool	-	true
Supervisions	String	no supervisions	true bei IDB of FB und false bei FB

Für DB- und FB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für DB- und FB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
IsIECCheckEnabled	Bool	false	false
IsRetainMemResEnabled[1]	Bool	false	false
MemoryReserve	Unsigned	0	false
RetainMemoryReserve[2]	Unsigned	0	false

[2] Wenn der Wert des Attributs "IsRetainMemResEnabled" "false" ist und das Attribut "RetainMemoryReserve" ungleich "0" ist, wird eine Ausnahme ausgelöst.

Für FB-, DB- und IDB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für FB-, DB- und IDB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
DownloadWithoutReinit	Bool	false	true

Für FB- und FC-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für FB- und FC-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
LibraryType	String	-	true
LibraryTypeVersionGuid	String	-	true

Für FB- und FC-Bausteine (AWL) geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für FB- und FC-Bausteine (AWL) exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
ParameterPassing	Bool	false	false

Für FB, FC und Instanz-DB von FB-Bausteinen geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für FB, FC und Instanz-DB von FB-Bausteinen exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
UDABlockProperties	String	""	false
UDAEnableTagReadback	Bool	false	false

Für Instanz-DBs von FBs und UDTs geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für Instanz-DBs von FB- und UDT-Bausteinen exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
InstanceOfName	String	""	false
InstanceOfNumber	Unsigned Short	-	true
InstanceOfType	enum BlockType	-	true
OfSystemLibElement	String	""	false
OfSystemLibVersion	String	""	false

Für OB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für OB-Bausteine für spezifische Plus PLCs exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
ApplicationCycle	Single	-	true
AutomaticMinimum	Bool	-	true
ConstantName	String	-	true
CycleTimeDistributedIO	Single	-	true
CyclicApplicationCycleTime	Single	-	true
CyclicTime	Int32	100000	true
DataExchangeMode	OBDataExchangeMode	Cyclic	true
DelayTime	Double	-	true
DistributedIOName	String	-	true
EnableTimeError	Bool	-	true

Attribut	Typ	Standardwert	Schreibgeschützt
EventClass	String	-	true
EventsToBeQueued	Int32	-	true
EventThresholdForTimeError	Int32	-	true
Execution	OBExecution	Never	true
Factor	Single	-	true
PhaseOffset	Int32	0	true
PriorityNumber	Int32	-	true
ProcessImagePartNumber	UInt32	-	true
ReportEvents	Bool	-	true
SecondaryType[3]	String	-	false
StartDate	DateTime	1/1/2012	true
SynchronousApplicationCycleTime	Single	-	true
TimeMode	OBTimeMode	System	true
TimeOfDay	DateTime	12:00 AM	true
TransformationDBNumber	UInt16	0xffff	true

[3] Beim Exportieren eines OB wird anhand der OB-Nummer zusätzlich der SecondaryType festgelegt. Die Zuordnung wird während des Importvorgangs geprüft. Ist die Zuordnung falsch, wird eine Ausnahme vom Typ „Recoverable“ ausgelöst.

Für FB-, FC- und OB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für FB-, FC- und OB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
HandleErrorsWithinBlock	Bool	false	true

Für FB-, FC- und UDT-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für FB-, FC- und UDT-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
LibraryConformanceStatus	String	-	false

Für GRAPH-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für GRAPH-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
AcknowledgeErrorsRequired	Bool	true	false
CreateMinimizedDB	Bool	false	false
ExtensionBlockName	String	-	-
GraphVersion	String	-	false
InitialValuesAcquisition	String	-	-
LanguageInNetworks	String	-	false
LockOperatingMode	Bool	false	false
PermanentILProcessingIn-MANMode	Bool	false	false
SkipSteps	Bool	false	false

Für GRAPH FB-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für GRAPH FB-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
WithAlarmHandling	Bool	true	false

Für SCL-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für SCL-Bausteine exportiert. Diese Attribute werden basierend auf dem Typ der PLCs exportiert.

Attribut	Typ	Standardwert	Schreibgeschützt
CheckArrayLimits	Bool	false	false
ExtendedStatus	Bool	false	false
DBAccessibleFromOPCUA	Bool	true	false

Für GRAPH-, SCL- und KOP/FUP-Bausteine geltende Attribute

Die folgenden Attribute werden mit den ausgewählten `ExportOptions.` für GRAPH-, SCL- und KOP/FUP-Bausteine exportiert:

Attribut	Typ	Standardwert	Schreibgeschützt
SetENOAutomatically	Bool	-	false

Programmcode

Um einen Baustein ohne Knowhow-Schutz in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a regular block
private static void ExportRegularBlock(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    plcBlock.Export(new FileInfo(string.Format(@"D:\Samples\{0}.xml", plcBlock.Name)),
ExportOptions.WithDefaults);
}
```

8.4.1.4 DBs mit Schnappschüssen exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Anwendung

Mit TIA Portal Openness können Sie die DBs mit Schnappschusswerten als XML exportieren und die Werte mit verschiedenen Schnappschusszeiten vergleichen. Mit dem Vergleichsergebnis können sie manuell einzelne Startwerte in der UI anpassen und für die spätere Wiederherstellung speichern.

Programmcode

Ändern Sie folgenden Programmcode, um Schnappschusswerte mit dem Snapshot Service zu exportieren:

```
InterfaceSnapshot interfaceSnapshot = dataBlock.GetService<InterfaceSnapshot>();
interfaceSnapshot.Export(new FileInfo("C:\temp\MyInterfaceSnapshot.xml"),
ExportOptions.None);
```

Der Snapshot Service "InterfaceSnapshot" wird im Namespace "Siemens.Engineering.SW.Blocks" bereitgestellt. Das Handling der Dateien (z. B. wenn das Exportverzeichnis nicht existiert; Erstellen des Exportverzeichnisses; wenn das Exportverzeichnis schreibgeschützt ist; wenn die Exportdatei bereits existiert) ist das gleiche

wie beim normalen Export über die Standardschnittstelle von Openness. Der Snapshot Service wird unterstützt für globale DBs, Instanz-DBs und Array-DBs.

Hinweis

Der Export von Schnappschusswerten mit dem Snapshot Service ist unabhängig vom Export über die Standardschnittstelle von Openness und beeinflusst daher nicht den bereits vorhandenen Export der Schnittstellenmitglieder. Die exportierte XML kann nicht mehr importiert werden.

Die Schnappschusswerte werden wie folgt exportiert:

```
<?xml version="1.0" encoding="utf-8"?>
<Document>
  <Engineering version="V15 SP1" />
  <DocumentInfo>
  ...
  </DocumentInfo>
  <SW.Blocks.InterfaceSnapshot ID="0">
   <AttributeList>
      <Name>GlobalDB</Name>
      <Snapshot ReadOnly="true"><SnapshotValues>
      <SnapshotValues>
      <Value Path="Static_1" Type="Bool">TRUE</Value>
      <Value Path="Static_2[0]" Type="Int">1</Value>
      <Value Path="Static_2[1]" Type="Int">2</Value>
      <Value Path="Static_2[2]" Type="Int">3</Value>
      <Value Path="Static_3" Type="DTL">DTL#1973-01-01-00:00:00</Value>
      <Value Path="Static_4.Element_1" Type="Int">7</Value>
      <Value Path="Static_4.Element_2[0]" Type="Bool">FALSE</Value>
      <Value Path="Static_4.Element_2[1]" Type="Bool">TRUE</Value>
      <Value Path="Static_4.Element_2[2]" Type="Bool">TRUE</Value>
      <Value Path="Static_4.Element_3.Element_1" Type="Int">5</Value>
      <Value Path="Static_4.Element_3.Element_2.Element_1" Type="Bool">TRUE</Value>
      <Value Path="Static_4.Element_3.Element_2.Element_2[0]" Type="Int">100</Value>
      <Value Path="Static_4.Element_3.Element_2.Element_2[1]" Type="Int">200</Value>
    </SnapshotValues></Snapshot>
    <SnapshotDate ReadOnly="true">2017-12-06T08:04:11.4590585Z</SnapshotDate>
    <StructureModified ReadOnly="true">2017-12-06T08:22:13.3292585Z</StructureModified>
   </AttributeList>
 </SW.Blocks.InterfaceSnapshot>
</Document>
```

Wenn ein DB keine Schnappschusswerte enthält, würde der Inhalt der exportierten Datei folgendermaßen aussehen:

```
<SnapshotValues xlmns="http://www.siemens.com/automation/Openness/SW/Interface/Snapshot/
v1"></SnapshotValues>
```

Siehe auch

Projekt öffnen (Seite 99)

8.4.1.5 Bausteine mit Know-how-Schutz exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)
- Die PLC ist nicht online.

Verwendung

Die resultierende XML-Datei ähnelt der Exportdatei eines Bausteins ohne Knowhow-Schutz. Der Export deckt jedoch nur die Daten der Benutzeroberfläche ab, die sichtbar sind, wenn der Baustein ohne Passwort geöffnet wird.

Die Attributliste des Bausteins zeigt an, dass der entsprechende Baustein Knowhow-Schutz besitzt.

Programmcode

Um die sichtbaren Daten eines Bausteins mit Knowhow-Schutz in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
private static void ExportBlock(PlcSoftware plcSoftware)
{
    PlcBlock plcBlock = plcSoftware.BlockGroup.Blocks.Find("MyBlock");
    plcBlock.Export(new FileInfo(string.Format(@”D:\Samples\{0}.xml”, plcBlock.Name)),
ExportOptions.WithDefaults);
}
```

8.4.1.6 Export/Import von SCL-Bausteinen

SCL-Anweisungen mit XML-Tags für Export

Die Exportoperation für SCL-Bausteine exportiert ihre entsprechenden XML-Tags auf der Grundlage des SCL-Anweisungstyps. Diese Operation unterstützt die SCL-Netzwerke von SCL-Anweisungen in KOP/FUP-Bausteinen von SCL-Anweisungen. Die SCL-Anweisungen sind klassifiziert als Textelemente, Operanden, Ausdrücke, Steueranweisungen usw. Die SCL-Bausteinanweisungen mit ihren entsprechenden exportierten XML-Tags und Attributen sind unten angegeben.

Neue Zeile

Neue Zeilen in SCL-Bausteinen werden durch NewLine XML-Tags dargestellt.

- Enthält das vorzeichenlose Attribut Num mit dem Standardwert 1.
- Attribut Num hat nicht den Wert 0.
- Nur bei SCL unterstützt.

SCL-Baustein	XML-Tag
	`<NewLine Num="2" />`

Leer

Leerzeichen in SCL-Bausteinen werden durch Blank XML-Tags dargestellt.

- Enthält das vorzeichenlose Attribut Num mit dem Standardwert 1.
- Attribut Num hat nicht den Wert 0.
- Nur bei SCL unterstützt.
- Unterstützt nicht das Attribut Integer, das in anderen Sprachen von STEP 7 verfügbar ist.

SCL-Baustein	XML-Tag
	`<Blank Num="2"/>`

Einrückung von SCL-Bausteinanweisungen

In den Einstellungen im TIA Portal können Sie die Einrückung von SCL-Code über Options/ Settings/General/Script/text editiors ändern. In der folgenden Tabelle wird die Art der Einrückung auf der Grundlage des Ident-Modus definiert.

Ident-Modus	Ergebnis
Ohne	Importoperation fügt die Leerzeichen ein, wie in den Quelldateien verfügbar.
Paragraph oder Smart	Importoperation fügt die angegebenen Ident-Leerzeichen in die importierte Datei ein.

Auf der Grundlage der gewählten Einrückung wird diese in der importierten XML file des SCL-Bausteins vorgenommen.

Kommentar

Ein- und mehrzeilige Kommentare in SCL-Bausteinen werden durch LineComment XML-Tags dargestellt.

- Nur das Tag LineComment (für einsprachigen Kommentar) wird von SCL unterstützt.
- Das Tag Comment (für mehrsprachigen Kommentar) wird von SCL nicht unterstützt.
- Enthält das Attribut Inserted mit Standardwert false.
- Inserted="false" zeigt "//" einzeiligen Kommentar in SCL-Bausteinen an.

- Inserted="true" zeigt "(**)" mehrzeiligen Kommentar in SCL-Bausteinen an.
- NoClosingBracket="true" zeigt Kommentare ohne schließende Klammer in SCL-Bausteinen an. Dieses Attribut ist optional und hat den Standardwert false.
- XML zeigt keine Kommentarhierarchie in SCL-Bausteinen an.

SCL-Baustein	XML-Tag

`// 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 >`
	Der verschachtelte Kommentar ist Bestandteil des äußeren Kommentartexts.

`(* comment without closing bracket`	`<LineComment Inserted="true"`
	`NoClosingBracket="true">`
	`  <Text> comment without`
	`closing bracket</Text>`
	`</LineComment >`

Region

Regionen in SCL-Bausteinen werden durch Token XML-Tags dargestellt.

- Das Text XML-Tag repräsentiert den Regionsnamen (region_name).
- Beim Attribut Text des Token XML-Tags spielt die Groß- und Kleinschreibung keine Rolle.
- Beim Importvorgang wird die Groß- und Kleinschreibung beachtet, und der Editor zeigt die Schlüsselwörter wie in den TIA Portal-Einstellungen konfiguriert an.
- Wenn das Schlüsselwort end_region mit ";" (Semikolon) im SCL-Baustein endet, steht das Zeichen ";" im Text XML-Tag.

SCL-Baustein	XML-Tag
region myregion ... end_region here is the end of myregion	<Token Text="REGION" /> <Blank /> <Text>myregion</Text> <NewLine /> ... <Token Text="END_REGION" /> <Blank /> <Text>here is the end of myregion</Text> <NewLine />
region // here are no blanks ... end_region	<Token Text="REGION" /> <NewLine /> <LineComment .../> <Token Text="END_REGION" /> <NewLine />
region ... end_region;	<Token Text="REGION" /> <NewLine /> ... <Token Text="END_REGION" /> <Text>;</Text> <NewLine />

Pragma

Pragma in SCL-Bausteinen werden durch Token XML-Tags dargestellt. Die Parameter werden im XML-Tag Access mit dem Attribut Scope als LiteralConstant dargestellt.

SCL-Baustein	XML-Tag
{PRAGMA_BEGIN 'Param1', 'Param2' (*parm 2*)} // something else {PRAGMA_END}	<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="}" />

Konstanten: Literale Konstanten

Die Konstanten in SCL-Bausteinen werden durch Access XML-Tags dargestellt.

- Das Attribut Scope kann Werte wie LiteralConstant, TypedConstant, LocalConstant, und GlobalConstant. haben.
- Die Namen von Konstanten, denen das Zeichen "#" vorangestellt ist, werden in XML ignoriert.
- Das Zeichen "#" wird während des Importvorgangs von XML hinzugefügt.
- Der Wert globaler Konstanten, repräsentiert durch Anführungszeichen, wird in XML ignoriert.
- Die Anführungszeichen werden während des Importvorgangs von XML hinzugefügt.

Art der Konstante	SCL-Baustein	XML-Tag

Literale Konstante: Integer	`#Out := 10;`	`<Access Scope="LiteralConstant">`
		`  <Constant>`
		`  <ConstantValue>10</ConstantValue>`
		`  <ConstantTypeInformative="true">LINT</ConstantType>`
		`  </Constant>`
		`</Access>`

Literale Konstante: String	`#myString := 'Hello world';`	`<Access Scope="LiteralConstant">`
		`  <Constant>`
		`  <ConstantValue>Hello world</ConstantValue>`
		
		`<ConstantTypeInformative="true">STRING</ConstantType>`
		`  </Constant>`
		`</Access>`

Literale Konstante: Typed	`#Out := int#10;`	`<Access Scope="TypedConstant">`
		`  <Constant>`
		`  <ConstantValue>int#10</ConstantValue>`
		`  </Constant>`
		`</Access>`

		Format von XML nach Export mit Einstellung ExportOptions.ReadOnly.
		
		`<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>`

Lokale Konstante	`#Out := #mylocal;`	`<Access Scope="LocalConstant">`
		`  <Constant Name="mylocal" />`
		`</Access>`

		Format von XML nach Export mit Einstellung ExportOptions.ReadOnly
		
		`<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>`

Art der Konstante	SCL-Baustein	XML-Tag

Globale Konstante	`#Out := "myglobal";`	`<Access Scope="GlobalConstant">`
		`  <Constant Name="myglobal" />`
		`</Access>`

		Format von XML nach Export mit Einstellung ExportOptions.ReadOnly.
		
		`<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>`

Die Adresskonstanten werden in SCL-Bausteinen nicht unterstützt und in dieser Tabelle ignoriert.

Variablen

Die lokalen und globalen Variablen in SCL-Bausteinen werden durch Access XML-Tags dargestellt.

- Das Attribut Scope hat Werte von LocalVariable und GlobalVariable.
- Das XML-Tag für das Zuweisen des Werts 10 wird hier ignoriert.

Art der Variable	SCL-Baustein	XML-Tag

Lokale Variable	`#Out := 10;`	`<Access Scope="LocalVariable">`
		`  <Symbol>`
		`  <Component Name="Out" />`
		`  </Symbol>`
		`</Access>`

Globale Variable	`"Tag_3":= 10;`	`<Access Scope="GlobalVariable">`
		`  <Symbol>`
		`  <Component Name="Tag_3" />`
		`  </Symbol>`
		`</Access>`

		Format von XML nach Export mit Einstellung ExportOptions.ReadOnly.
		
		`<Access Scope="GlobalVariable">`
		`  <Symbol>`
		`  <Component Name="Tag_3" />`
		`  <Address Area="Memory" Type="Int" BitOffset="96" Informative="true" />`
		`  </Symbol>`
		`</Access>`

Ausdrücke

Die einfachen Ausdrücke in SCL-Bausteinen werden durch Access XML-Tags dargestellt. Das Attribut Scope hat den Wert von LocalVariable für die Ausdrücke.

SCL-Baustein	XML-Tag

`#a := #b + #c;`	`<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=";" />`

Steuerstrukturen in SCL-Bausteinen

Die Steueranweisungen wie IF, CASE, FOR, WHILE, REPEAT, GOTO, EXIT, CONTINE, and RETURN werden durch Token XML -Tags dargestellt.

- Die im SCL-Baustein verwendeten bedingten Symbole wie >, <, & werden in XML als Escape-Sequenzen (< > &amp) dargestellt.
- Diese Kombination von XML-Tags ist nur bei SCL-Bausteinen gültig. In anderen Sprachen wird eine Ausnahme ausgelöst.

Name des Baust-eins	SCL-Baustein	XML-Tag
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=";" />

Name des Baust-eins	SCL-Baustein	XML-Tag

CASE	CASE #a OF
	  1 (*test*): // Statement section case 1
	  ;
	  2..4: // Statement section case 2 to 4
	  ;
	  ELSE // Statement section ELSE
	  ;
	END_CASE;
		<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 >

Name des Baust-eins	SCL-Baustein	XML-Tag
		<NewLine /> <Blank Num="4"/> <Token Text=";" /> <NewLine /> <Blank Num="2"/> <Token Text="ELSE" /> <NewLine /> <Blank Num="4"/> <Token Text=";" /> <NewLine /> <Token Text="END_CASE" /> <Token Text=";" />

Name des Baust-eins	SCL-Baustein	XML-Tag

FOR	FOR #i := #a TO #b DO
	  // Statement section FOR
	  ;
	END_FOR;
		<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=";" />

Name des Baust-eins	SCL-Baustein	XML-Tag

WHILE	WHILE #a<#b DO
	  // Statement section WHILE
	  ;
	END_WHILE;
		<Token Text="WHILE" />
		<Blank />
		<Access Scope="LocalVariable">
		  <Symbol>
		  <Component Name="a" />
		  </Symbol>
		</Access>
		<Token Text="<" />
		<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=";" />

Name des Baust-eins	SCL-Baustein	XML-Tag

REPEAT	REPEAT
	  // Statement section REPEAT
	  ;
	UNTIL #a<#b END_REPEAT;
		<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="<" />
		<Access Scope="LocalVariable">
		  <Symbol>
		  <Component Name="b" />
		  </Symbol>
		</Access>
		<Blank />
		<Token Text="END_REPEAT" />
		<Token Text=";" />

Name des Baust-eins	SCL-Baustein	XML-Tag
GOTO	here // well : // this is goto statement	XML-Beispiel für Definition von GOTO-Labels <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>
	GOTO (*comment*) here;	XML-Beispiel für Verwendung von GOTO-Labels <Token Text="GOTO" /> <Blank /> <LineComment inserted=”true”> <Text>comment</Text> </LineComment> <Blank /> <Access Scope="Label"> <Label Name="here" /> </Access> <Token Text=";" />

Referenzierende Attribute

Die referenzierenden Attribute von SCL-Bausteinen werden durch das Attribut AccessModifier des Tags Component dargestellt.

- Zum einfachen Referenzieren hat AccessModifer den Wert als Reference.
- Zur Array-Referenzierung hat AccessModifier den Wert als ReferenceToArray..

SCL-Baustein	XML-Tag

`RefToUDT^(*RefToUDT*).element`	`<Symbol>`
	`  <Component Name="RefToUDT"`
	`AccessModifier="Reference" />`
	`  <Token Text="^" />`
	`  <LineComment Inserted="True">`
	`  <Text>RefToUDT</Text>`
	`  </LineComment>`
	`  <Token Text="." />`
	`  <Component Name="element" />`
	`</Symbol>`

RefToArrayOfUDT^(*RefToArrayOfUDT*)[#i].element	`<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>`

8.4.1.7 Export/Import strukturierter Typen von SCL-Bausteinen

Strukturierte SCL-Typen mit XML-Tags für den Export

In die strukturierten SCL-Typen können Sie Leerzeichen, neue Zeilen und Kommentare zu den SCL-Anweisungen einfügen. Die strukturierten SCL-Anweisungen mit ihren entsprechenden exportierten XML-Tags und Attributen sind unten angegeben.

Globaler Zugriff

In SCL-Anweisungen stehen Variablen und Konstanten für globalen Zugriff in Anführungszeichen. Die Kommentare zwischen den Variablen und Adressteilen werden durch das LineComment XML-Tag dargestellt.

<table>
<tr><th>SCL-Baustein</th><th>XML-Tag</th></tr>
<tr><td>"Datenbaustein_1".(*Kommentare 1*)Statisch_1(*Kommentare 2*).Statisch_2</td><td><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></td></tr>
<tr><td rowspan="2">"Data_block_1".Static_1 := 10</td><td>Format von XML nach Export mit Einstellung ExportOptions.None

<pre>
<Access Scope="GlobalVariable">
    <Symbol>
        <Component
Name="Data_block_1" />
         <Token Text="." />
         <Component Name="Static_1" /
>
    </Symbol>
</Access>
</pre></td></tr>
<tr><td>Format von XML nach Export mit Einstellung ExportOptions.ReadOnly

<pre>
<Access Scope="GlobalVariable">
    <Symbol>
        <Component
Name="Data_block_1" />
         <Token Text="." />
         <Component Name="Static_1" /
>
         <Address Area="DB"
Type="Word" BlockNumber="1"
BitOffset="0" Informative="true" />
    </Symbol>
</Access>
</pre></td></tr>
</table>

Verwendung von Anführungszeichen und

Die in der ersten Ebene verwendeten Anführungszeichen beschreiben die Art der Variable und dienen als Escape-Sequenzen für Sonderzeichen in SCL-Anweisungen. Wenn Anführungszeichen in der ersten Ebene verwendet werden, definieren sie die Variable als globale Variable. Wenn die Anführungszeichen nach # verwendet werden, stellen sie die Escape-Sequenz von Sonderzeichen wie # und von Leerzeichen dar.

- Zum Darstellen der unterschiedlichen Verwendung dient in der XML-Datei das Tag BooleanAttributes mit dem Attribut Name. Der Name enthält Werte wie HasQuotes und HasHash.
- Um im Scope-Attribut die Struktur zu definieren, wird # festgelegt.
- Diese Werte gelten nur für SCL.
- Die Standardwerte für diese Tags sind anfänglich FALSE, aber die Werte werden auch mit der Einstellung ExportOptions.WithDefaults nie exportiert.

SCL-Baustein	XML-Tag

`"a".#b."c".#"d"`	`<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 />`

Array

SCL erlaubt das Einfügen von Kommentaren in die Arrayindizes um "[" und "]". Zum Kennzeichnen des Vorhandenseins eines Arrays dient in der XML-Datei das Attribut AccessModifier im Tag Component.

- Wenn Accessmodifier den Wert Array enthält, ist ein untergeordnetes Tag Access obligatorisch, um die Indexvariable des Arrays anzuzeigen.
- Der Standardwert für AccessModifier ist None.

SCL-Baustein	XML-Tag

`#a.b[#i+#j,#k+#l].c`	`<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>`

Absoluter Zugriff

SCL ermöglicht verschiedene Zugriffsarten wie Absolut, Absolut-Offset, Gemischt (Datenbank und Membervariable), Slice, Peripher und Direkt. Der absolute Zugriff wird in XML durch das Tag Address dargestellt.

- Das Zeichen % in der DB wird in XML nicht geschrieben. Es wird beim Import automatisch erzeugt.
- Zwischen den Teilen der Adresse sind Leerzeichen zulässig.

SCL-Baustein	XML-Tag

`%DB20 . DBW10`	`<Access Scope="Address">`
	`  <Symbol>`
	`  <Address Area="DB" BlockNumber="20" />`
	`  <Blank />`
	`  <Token Text="." />`
	`  <Blank />`
	`  <Address Area="DB" BitOffset="80" Type="Word"/>`
	`  </Symbol>`
	`</Access>`

`%DB20.DBX10.3 := true;`	Folgende XML gilt für alle Sprachen außer SCL.
	
	`<Access Scope="Address">`
	`<Address Area="DB" BlockNumber="20" BitOffset="83" Type="Bool" />`
	`</Access>`
	
	Folgende XML gilt für SCL.
	
	`<Access Scope="Address">`
	`  <Symbol>`
	`  <Address Area="DB" BlockNumber="20" />`
	`  <Token Text="." />`
	`  <Address Area="DB" BitOffset="83" Type="Bool"/>`
	`  </Symbol>`
	`</Access>`

Zugriffsart "Absolut-Offset"

In AWL stellt das Tag AbsoluteOffset die Zugriffsart "Absolut-Offset" dar. In SCL wird das Tag Address für den absoluten Zugriff verwendet.

SCL-Baustein	XML-Tag

`#Input_DB_ANY.%DBX2.3 := TRUE;`	`<Access Scope="LocalVariable">`
	`  <Symbol>`
	`  <Component Name="Input_DB_ANY" />`
	`  <Token Name="." />`
	`  <Address BitOffset="19" Type="Bool" />`
	`  </Symbol>`
	`</Access>`

Zugriffsart "Slice"

In SCL wird das Attribut SliceAccessModifier nicht unterstützt, und die Zugriffsart "Slice" wird durch das Tag Token dargestellt.

SCL-Baustein	XML-Tag

`"tag_1"(*1*).(*2*)member(*3*).(*4*) %x1`	`<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>`

Peripherer Zugriff

Der periphere Zugriff wird durch das Tag Token dargestellt.

SCL-Baustein	XML-Tag

`"tag_1"(*1*).(*2*)member:P`	`<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>`

Zugriffsart "Direkt"

Die Anweisungen TypeOf und TypeOfDB werden entweder mit systemdefiniertem oder benutzerdefiertem Typ ausgeführt. Angegeben sind die Typen im Tag Access mit dem Attribut Scope und den Werten SystemType und UserType.

SCL-Baustein	XML-Tag

Beispiel für systemdefinierten Typ
`if TypeOf( #inVariant ) = TO_SpeedAxis then … end_if`
	`<Token text="=" />`
	`</Blank>`
	`<Access Scope="SystemType">`
	`  <DataType>TO_SpeedAxis</DataType>`
	`</Access>`

Beispiel für benutzerdefinierten Typ
`if TypeOf( #inVariant ) = "aUserDefinedType" then … end_if`
	`<Token text="=" />`
	`</Blank>`
	`<Access Scope="UserType">`
	`  <DataType>aUserDefinedType</DataType>`
	`</Access>`

8.4.1.8 Export/Import von SCL-Aufrufbausteinen

SCL-Aufrufbausteine mit XML-Tags für den Export

SCL-Aufrufparameter werden in XML durch das Tag Parameter dargestellt. Das Attribut informative dient zum Darstellen der nicht zugewiesenen Parameter und Rückgabewerte wie Zeitstempel, Merkerinformationen usw. Das XML-Format folgt der gleichen willkürlichen Reihenfolge wie im SCL-Baustein.

Nachstehend ein Beispiel für einen Bausteinaufruf:

SCL-Baustein	XML-Tag

`#Callee_Instance(Input_1 := 5);`	Format von XML nach Export mit Einstellung ExportOptions.None
	
	`<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=";" />`

	Format von XML nach Export mit Einstellung ExportOptions.ReadOnly
	
	`<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=":=" />`

SCL-Baustein	XML-Tag
	<Blank /> <Access Scope="LiteralConstant"> <Constant> <ConstantType>Int</ConstantType> <ConstantValue>5</ConstantValue> </Constant> </Access> </Parameter> <Token text=")" /> </CallInfo> </Access> <Token text=";" />

Beispiel für unverbundene Parameter

Der FB hat vier Parameter, wobei a, b, c sowie d. b und d nicht verbunden sind.

SCL-Baustein	XML-Tag
"Block_4_DB"(a:=TRUE,c:=TRUE);	<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>

Beispiel für "ein Parameter"

Der SCL-Baustein ermöglicht Ihnen das Weglassen des Parameternamens. Dieser Parameter wird durch das Tag NamelessParameter dargestellt. Das Tag NamelessParameter hat keine Attribute und gilt nur für SCL.

SCL-Baustein	XML-Tag

`"Block_4_DB"(TRUE);`	`<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>`

Ausdruck als Aktualparameter

SCL-Baustein	XML-Tag

`#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=";" />`

Ausdruck als Aktualparameter ohne Formalparameter

SCL-Baustein	XML-Tag
`#Callee_Instance(#a+3);`	`<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=";" />

Funktionsaufruf

SCL-Baustein	XML-Tag
`#myInt := "MyFunction"(Param_1 := 1, Param_2 := 15, Param_3 := TRUE);`	`<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">   ...`

Absoluter Aufruf

In SCL kann der Aufruf über die absolute Adresse des DB initiiert werden. Wegen der absoluten Adresse ist das Attribut Name des Knotens CallInfo leer.

Eine wiederherstellbare Ausnahme wird durch den Import ausgelöst, wenn

- Ein Knoten "Address" mit gültigem Wert des Attributs Name verfügbar ist.
- Der Knoten "Address" nicht vorhanden ist und kein gültiger Wert des Attributs Name vorhanden ist.

SCL-Baustein	XML-Tag

`%DB20(...);`	`<Access Scope="Call">`
	`  <CallInfo Name="" BlockType="FB">`
	`  <Instance Scope="GlobalVariable">`
	`  <Address Area="DB" BlockNumber="20" />`
	`  </Instance>`
	`  <Token text="(" />`
	`  <Parameter>`
	`  …`
	`  </Parameter>`
	`  <Token text=")" />`
	`  </CallInfo>`
	`</Access>`

Anweisung

Die Anweisung im SCL-Baustein wird während des Importvorgangs in der Systembibliothek geprüft, und die Anweisungsversionen werden beim Exportvorgang nicht exportiert.

Der allgemeine Anweisungstyp ist unten angegeben.

SCL-Baustein	XML-Tag

`#myInt := ATTACH(OB_NR := 1, EVENT := 15, ADD := TRUE);`	Format von XML nach Export mit Einstellung ExportOptions.ReadOnly
	
	`<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>`

SCL-Baustein	XML-Tag
	</Constant> </Access> </Parameter> <Parameter Name="RET_VAL" Informative="true" /> <Token text=")" /> </Instruction> </Access> <Token text=";" />

Anweisung mit Vorlage

Wenn der Vorlagenparameter den Anweisungsnamen ergänzt, ist der Export des Vorlagenparameters notwendig. Wenn ein "TemplateValue" -Tag mit Attribut Type="Type" auf das Instruction-Tag folgt, verkettet der Importvorgang den Vorlagenwert mit dem Anweisungsnamen.

SCL-Baustein	XML-Tag

`"tag_4" := MIN_DINT( IN1:="Tag_1", IN2:="Tag_2", IN3:="Tag_3" );`	`<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>`
	`...`

Umwandlung

Bei Umwandlungsfunktionen werden der reale Anweisungsname und seine Vorlage nicht exportiert. Stattdessen wird der im SCL-Baustein verwendete Name exportiert.

SCL-Baustein	XML-Tag
`#output_1 := TIME_TO_S5TIME(#input_1);`	`<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>...`

Anweisung mit Instanz

Instanz und Anweisung werden durch Leerzeichen getrennt. Leerzeichen sind optional, und sie können durch neue Zeilen und Kommentare dargestellt werden. Die Anweisung TON wird durch das Attribut Name des Tags Instruction dargestellt.

SCL-Baustein	XML-Tag

`IEC_Timer_0_DB . TON (IN:="Tag_1", PT:="Tag_2");`	`<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=";" />`

Alarmkonstante

Die Alarmkonstanten werden nur in S7 400-PLCs verwendet, und der exportierte XML-Code hat Ähnlichkeit mit anderen Sprachen.

SCL-Baustein	XML-Tag
"Block_1_DB"(16#0000_0001);	Format von XML nach Export mit Einstellung ExportOptions.None <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>
	Format von XML nach Export mit Einstellung ExportOptions.ReadOnly <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>

ENO (Freigabeausgang)

Um den Freigabeausgang ENO im SCL-Baustein zu unterstützen, wird im Tag "Access" das Attribut "Scope" mit dem Wert "PredefinedVariable" verwendet. Es enthält auch das Tag "PredefinedVariable" als untergeordnetes Element des Tags Access.

- Das Tag "PredefinedVariable" hat ein obligatorisches Attribut "Name".
- Der Umfang "PredefinedVariable" und das Tag "PredefinedVariable" sind nur für SCL zulässig.

SCL-Baustein	XML-Tag

`Call(…, ENO => ENO);`	`<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=";" />`

`IF ENO = #c THEN …`	`<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" />`

8.4.1.9 F-Bausteine exportieren

Fehlersichere Bausteine exportieren

Fehlersichere Bausteine werden wie Standardbausteine exportiert. Bei fehlersicheren Bausteinen beginnt der Wert des Attributs "`ProgrammingLanguage`" mit einem Präfix "`F_`".

Hinweis

Der Import einer Datei ist nicht möglich, wenn der Wert des Attributs "`ProgrammingLanguage`" mit einem Präfix "`F_`" beginnt.

Fehlersichere Bausteine als Standardbausteine importieren

Fehlersichere Bausteine können als Standardbausteine importiert werden, wenn das Präfix "`F_`" vom Wert aller Attribute "`ProgrammingLanguage`" entfernt wird.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

Bausteine exportieren (Seite 501)

Bausteine mit Know-how-Schutz exportieren (Seite 509)

8.4.1.10 System-Bausteine exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)
- Das Projekt enthält einen Systembaustein.
- Die PLC ist nicht online.

Verwendung

Nur sichtbare Systembausteine sind in der Bausteinzusammensetzung verfügbar, also keine SFBs und SFCs. Die resultierende XML-Datei ähnelt der Exportdatei eines Bausteins.

Programmcode

Um die sichtbaren Daten eines Bausteins in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports system blocks
private static void ExportSystemBlocks(PlcSoftware plcsoftware)
{
    PlcSystemBlockGroup sbSystemGroup = plcsoftware.BlockGroup.SystemBlockGroups[0];
    foreach (PlcSystemBlockGroup group in sbSystemGroup.Groups)
    {
        foreach (PlcBlock block in group.Blocks)
        {
            block.Export(new FileInfo(string.Format(@"D:\Samples\{0}.xml", block.Name)),
ExportOptions.WithDefaults);
        }
    }
}
```

8.4.1.11 GRAPH-Bausteine mit mehrsprachigem Text exportieren

XML-Struktur von GRAPH-Bausteinen mit mehrsprachigem Text

Die Export-XML von GRAPH-Bausteinen enthält die übersetzen Schrittnamen und Transitionsnamen des Graphen. Diese übersetzten mehrsprachigen Texte werden jeweils unter dem übergeordneten Element Step und Transition als Elemente StepName und TransitionName dargestellt. Diese Elemente enthalten für jede unterstützte Sprache ein Element MultiLanguageText. Die Texte für die Sprachen, die nicht explizit eingestellt sind, werden nicht exportiert. Wenn keine Übersetzung vorhanden ist, werden die Elemente StepName und TransitionName nicht exportiert. Die Elemente StepName und TransitionName sind optional. Die TIA Portal Openness XML-Importoperation löst bei den Graph-Versionen kleiner als V5.0 eine wiederherstellbare Ausnahme aus.

Beispiel für das Element StepName

```
<Steps>
 <Step Number="1" Init="true" Name="Step1" MaximumStepTime="T#10S" WarningTime="T#7S">
   <StepName>
      <MultiLanguageText Lang="de-DE">stepDE</MultiLanguageText>
      <MultiLanguageText Lang="en-US">stepEN</MultiLanguageText>
      <MultiLanguageText Lang="it-CH">stepIT</MultiLanguageText>
   </StepName>
      ..
 </Step>
..
</Steps>
```

Beispiel für das Element TransitionName

```
<Transitions>
  <Transition IsMissing="false" Name="Trans1" Number="1" ProgrammingLanguage="LAD">
   <TransitionName>
      <MultiLanguageText Lang="de-DE">transDE</MultiLanguageText>
      <MultiLanguageText Lang="en-US">transEN</MultiLanguageText>
      <MultiLanguageText Lang="it-CH">transIT</MultiLanguageText>
   </TransitionName>
      ..
 </Transition>
..
</Transitions>
```

8.4.1.12 Baustein importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Verwendung

Die TIA Portal Openness API unterstützt den Import von Bausteinen mit den Programmiersprachen "AWL", "FUP", "GRAPH", "SCL" oder "KOP" aus einer XML-Datei. Die folgenden Bausteintypen werden unterstützt:

- Funktionsbausteine (FB)
- Funktionen (FC)
- Organisationsbausteine (OB)
- Globale Datenbausteine (DB)

Hinweis

Importieren optimierter Datenbausteine

Optimierte Datenbausteine werden nur von CPUs ab S7-1200 unterstützt. Wenn Sie optimierte Datenbausteine in S7-300 oder S7-400 importieren, wird eine Ausnahme ausgelöst und der Import schlägt fehl.

Reaktion auf den Import

Beim Importieren eines Bausteins gelten die folgenden Regeln:

- Die XML-Datei kann weniger Daten enthalten als der Baustein im Projekt, z. B. weniger Parameter.
- Redundante Informationen wie Aufrufinformationen müssen im Projekt und in der XML-Datei identisch sein. Andernfalls wird eine Ausnahme ausgelöst.
- Die Daten in der XML-Datei dürfen im Hinblick auf ihre Übersetzungsfähigkeit im TIA Portal „inkonsistent“ sein.
- Attribute mit den Attributen "ReadOnly=True" und "Informative=True" werden nicht importiert.
- Fehlende Instanz-DBs werden nicht automatisch erstellt.
- Wenn in der XML-Datei keine Bausteinnummer angegeben ist, wird die Bausteinnummer automatisch zugeordnet.
- Wenn der Baustein im Projekt nicht vorhanden ist und in der XML-Datei keine Versionsinformationen angegeben sind, wird die Version "0.1" zugeordnet.

Programmcode

Ändern Sie folgenden Programmcode:

```
//Import blocks
private static void ImportBlocks(PlcSoftware plcSoftware)
{
    PlcBlockGroup blockGroup = plcSoftware.BlockGroup;
    IList<PlcBlock> blocks = blockGroup.Blocks.Import(new FileInfo(@"D:\Blocks
\myBlock.xml"), ImportOptions.Override);
}
```

Ändern Sie folgenden Programmcode:

```
//Import system blocks
private static void ImportSystemBlocks(PlcSoftware plcSoftware)
{
    PlcBlockSystemGroup systemblockGroup = plcSoftware.BlockGroup;
    IList<PlcBlock> blocks = systemblockGroup.Blocks.Import(new FileInfo(@"D:\Blocks
\myBlock.xml"), ImportOptions.Override);
}
```

8.4.1.13 Bausteine/UDT mit offenem Verweis importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)
- Der PLC ist nicht online.

Anwendung

Mit der Openness API können Sie bei STEP 7-Objekten einen neuen Importmodus verwenden, bei dem Sie Bausteine und UDTs auch dann importieren können, wenn ein zugehöriges Objekt fehlt.

Die Openness-Schnittstelle unterstützt den neuen Importmodus unter den folgenden Bedingungen:

Import von	Objektverweis
UDT	UDT
DB (global)	UDT
IDBofUDT	UDT
IDBofFB	FB
ArrayDB	Array von UDT
FB	UDT (Schnittstelle), Multiinstanz
FC	UDT (Schnittstelle)

Programmcode

Den neuen Modus können Sie dank einer neuen Überladung der entsprechenden Import-Methode verwenden. Die neue Überladung weist einen zusätzlichen Parameter auf, der einen Wert der neuen markierten Enumeration SWImportOptions akzeptiert. Um den Import zu ermöglichen, können Sie SWImportOptions.IgnoreMissingReferencedObject auch dann verwenden, wenn das Objekt des Verweises fehlt.

```
Flagged Enum SWImportOptions
 {
 None = 0,
 IgnoreStructuralChanges = 1,
 IgnoreMissingReferencedObjects = 2
 }
... // All kinds of blocks
PlcBlockComposition.Import(file, ImportOptions.None,
SWImportOptions.IgnoreMissingReferencedObject);
...
... // UDTs
PlcTypeComposition.Import(file, ImportOptions.None,
SWImportOptions.IgnoreMissingReferencedObject);
...
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

8.4.1.14 Bausteine/UDT für Strukturänderungsobjekte importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)
- Der PLC ist nicht online.

Anwendung

Mit der Openness API können Sie Bausteine und UDTs auch dann importieren, wenn aufgrund einer Strukturänderung an zugehörigen Objekten Instanzdaten verloren gegangen sind.

Die Openness-Schnittstelle unterstützt den neuen Importmodus unter den folgenden Bedingungen:

Import von	Objektverweisen
Variable	UDT
UDT	UDT
DB (global)	UDT
IDBofUDT	UDT
IDBofFB	FB
ArrayDB	Array von UDT
FB	UDT (Schnittstelle), Multiinstanz
FC	UDT (Schnittstelle)

Programmcode

Den neuen Modus können Sie dank einer neuen Überladung der entsprechenden Import - Methode verwenden. Die neue Überladung weist einen zusätzlichen Parameter auf, der einen Wert der neuen markierten Enumeration SWImportOptions akzeptiert. Um den Import zu ermöglichen, können Sie SWImportOptions.IgnoreStructuralChanges auch in Fällen von Strukturänderungen und Datenverlust verwenden.

```
Flagged Enum SWImportOptions
{
None = 0,
IgnoreStructuralChanges = 1,
IgnoreMissingReferencedObjects = 2
}
...
// All kinds of blocks
PlcBlockComposition.Import(file, ImportOptions.None,
SWImportOptions.IgnoreStructuralChanges);
...
...
// UDTs
PlcTypeComposition.Import(file, ImportOptions.None,
SWImportOptions.IgnoreStructuralChanges);
...
```

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

8.4.2 Variablentabellen

8.4.2.1 PLC-Variablentabellen exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Verwendung

Pro PLC-Variablentabelle wird eine XML-Datei exportiert.

Die Schnittstelle TIA Portal Openness API unterstützt den Export aller PLC-Variablentabellen aus der Systemgruppe und deren Untergruppen.

Programmcode

Ändern Sie den folgenden Programmcode, um alle PLC-Variablentabellen aus der Systemgruppe und deren Untergruppen zu exportieren:

```
private static void ExportAllTagTables(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    // Export all tables in the system group
    ExportTagTables(plcTagTableSystemGroup.TagTables);
    // Export the tables in underlying user groups
    foreach(PlcTagTableUserGroup userGroup in plcTagTableSystemGroup.Groups)
    {
        ExportUserGroupDeep(userGroup);
    }
}

private static void ExportTagTables(PlcTagTableComposition tagTables)
{
    foreach(PlcTagTable table in tagTables)
    {
        table.Export(new FileInfo(string.Format(@"D:\Samples\{0}.xml", table.Name)),
ExportOptions.WithDefaults);
    }
}

private static void ExportUserGroupDeep(PlcTagTableUserGroup group)
{
    ExportTagTables(group.TagTables);
    foreach(PlcTagTableUserGroup userGroup in group.Groups)
    {
        ExportUserGroupDeep(userGroup);
    }
}
```

Siehe auch

Export von Projektierungsdaten (Seite 427)

8.4.2.2 PLC-Variablentabelle importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Programmcode

Ändern Sie den folgenden Programmcode, um PLC-Variablentabellen oder eine Ordnerstruktur mit PLC-Variablentabellen aus einer XML-Datei in die Systemgruppe oder eine benutzerdefinierte Gruppe zu importieren:

```
//Imports tag tables to the tag system group
private static void ImportTagTable(PlcSoftware plcSoftware)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    PlcTagTableComposition tagTables = plcTagTableSystemGroup.TagTables;
    tagTables.Import(new FileInfo(@"D:\Samples\myTagTable.xml"), ImportOptions.Override);
    // Or, to import into a subfolder:
    // plcTagTableSystemGroup.Groups.Find("SubGroup").TagTables.Import(new FileInfo(@"D:
\Samples\myTagTable.xml"), ImportOptions.Override);
}
```

Siehe auch

Anmerkungen zur Leistung von TIA Portal Openness (Seite 42)

8.4.2.3 Einzelne Variable oder Konstante aus einer PLC-Variablentabelle exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Verwendung

Die API-Schnittstelle unterstützt den Export einer Variablen oder Konstanten aus einer PLC-Variablentabelle in eine XML-Datei. Achten Sie dabei darauf, dass die verwendeten Variablentabellennamen den Dateibenennungskonventionen Ihres Dateisystems entsprechen.

Der Kommentar einer Variablen oder Konstanten wird nur exportiert, wenn mindestens eine Sprache für den Kommentar festgelegt ist. Wenn der Kommentar nicht für alle Projektsprachen festgelegt ist, wird dieser Kommentar nur für die festgelegten Projektsprachen exportiert.

Hinweis

PLC-Systemkonstanten

PLC-Systemkonstanten werden vom Export und Import ausgeschlossen.

Programmcode

Um eine bestimmte Variable oder Konstante aus einer PLC-Variablentabelle in eine XML-Datei zu exportieren, ändern Sie folgenden Programmcode:

```
//Exports a single tag or constant of a controller tag table
private static void ExportTag(PlcSoftware plcSoftware, string tagName)
{
    PlcTagTableSystemGroup plcTagTableSystemGroup = plcSoftware.TagTableGroup;
    PlcTag tag = plcTagTableSystemGroup.TagTables[0].Tags.Find(tagName);
    if(tag 0= 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);
}
```

Siehe auch

Export von Projektierungsdaten (Seite 427)

Anmerkungen zur Leistung von TIA Portal Openness (Seite 42)

8.4.2.4 Einzelne Variable oder Konstante in eine PLC-Variablentabelle importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts (Seite 99)

Verwendung

Sie können in einem Importaufruf entweder Variablen oder Konstanten importieren.

Hinweis

Konstanten können nur als Anwenderkonstanten importiert werden.

Programmcode

Ändern Sie den folgenden Programmcode, um Variablengruppen oder einzelne Variablen und Konstanten aus einer XML-Datei zu importieren:

```
//Imports tags into a plc tag table
private static void ImportTag(PlcSoftware plcSoftware, string tagtableName)
{
    PlcTagTableSystemGroup plcTagTableSystemgroup = plcSoftware.TagTableGroup;
    PlcTagTable tagTable = plcTagTableSystemgroup.TagTables.Find(tagtableName);
    if(tagTable == null) return;

    tagTable.Tags.Import(new FileInfo(@"D:\Samples\myTags.xml"), ImportOptions.Override);
}

//Imports constants into a plc tag table
private static void ImportConstant(PlcSoftware plcSoftware, string tagtableName)
{
    PlcTagTableSystemGroup plcTagTableSystemgroup = plcSoftware.TagTableGroup;
    PlcTagTable tagTable = plcTagTableSystemgroup.TagTables.Find(tagtableName);
    if(tagTable == null) return;

    tagTable.UserConstants.Import(new FileInfo(@"D:\Samples\myConstants.xml"),
ImportOptions.Override);
}
```

Siehe auch

Export von Projektierungsdaten (Seite 427)

Anmerkungen zur Leistung von TIA Portal Openness (Seite 42)

8.4.3 Anwenderdatentyp exportieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Programmcode

Ändern Sie den folgenden Programmcode, um einen Anwenderdatentyp in eine XML-Datei zu exportieren:

```
//Exports a user defined type
private static void ExportUserDefinedType(PlcSoftware plcSoftware)
{
    string udtname = "udt name XYZ";
    PlcTypeComposition types = plcSoftware.TypeGroup.Types;
    PlcType udt = types.Find(udtname);
    udt.Export(new FileInfo(string.Format(@"C:\OpennessSamples\udts\{0}.xml", udt.Name)),
ExportOptions.WithDefaults);
}
```

8.4.4 Anwenderdatentyp importieren

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Verwendung

Die API-Schnittstelle unterstützt das Importieren von Anwenderdatentypen aus einer XML-Datei.

Syntax der Importdatei

Der folgende Beispielcode zeigt einen Ausschnitt aus einer Importdatei eines benutzerdefinierten Datentyps:

```
<Section Name="Input">
 <Member Name="Input1" Datatype=quot;myudt1&quot;>
  <Sections>
   <Section Name="None">
    <Member Name="MyUDT1Member1" Datatype="bool"/>
    <Member Name="MyUDT1Member2" Datatype=&quot;myudt1&quot;>
     <Sections...
```

Hinweis

Syntax für benutzerdefinierte Datentypen von Elementen

Wenn der benutzerdefinierte Datentyp eines Elements in der Importdatei für Anwenderdatentypen eine falsche Syntax aufweist, wird eine Ausnahme ausgelöst.

Achten Sie darauf, dass benutzerdefinierte Datentypen mit `&quot;` notiert werden.

Programmcode

Um einen Anwenderdatentyp zu importieren, ändern Sie folgenden Programmcode:

```
//Imports user data type
private static void ImportUserDataType(PlcSoftware plcSoftware)
{
    FileInfo fullFilePath = new FileInfo(@"C:\OpennessSamples\Import\ExportedPlcType.xml");
    PlcTypeComposition types = plcSoftware.TypeGroup.Types;
    IList<PlcType> importedTypes = types.Import(fullFilePath, ImportOptions.Override);
}
```

Siehe auch

Import von Projektierungsdaten (Seite 429)

8.4.5 Export von Daten im Format OPC UA XML

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist nicht online.

Verwendung

Sie können PLC-Daten als Datei im Format OPC UA XML exportieren, indem Sie eine Aktion auf die TIA Portal Openness API aufrufen. Als Eingabeparameter für die Aktion benötigen Sie einen absoluten Verzeichnispfad, in dem die XML-Datei gespeichert wird.

Programmcode

Ändern Sie folgenden Programmcode.

```
//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 Hardware-Daten importieren/exportieren

8.5.1 AML-Dateiformat

Einleitung

AutomationML ist ein neutrales, auf XML basierendes Datenformat zum Speichern und Austauschen von Engineering-Informationen einer Anlage und wird als offener Standard angeboten. Ziel von AutomationML ist die Verschaltung der heterogenen Werkzeuglandschaft moderner Engineering-Tools in deren unterschiedlichen Disziplinen wie mechanisches Anlagen-Engineering, Auslegung der Elektrik, HMI, PLC, Robotersteuerung.

Das für den Export und Import von CAx-Daten verwendete Klassifizierungsmodell basiert auf den folgenden AML-Standards:

- Whitepaper AutomationML Teil 1 – AutomationML Architecture, Oktober 2014
- Whitepaper AutomationML Teil 2 – AutomationML Role Class Libraries, Oktober 2014
- Whitepaper AutomationML Teil 4 – AutomationML Logic, Mai 2010
- Whitepaper AutomationML – AutomationML Communication, September 2014
- Whitepaper AutomationML – AutomationML and eCl@ss Integration, November 2015
- Anwendungsempfehlungen: Automation Project Configuration - AR_001E Version 1.0.0, Mai 2017

Schema

Das AutomationML-Datenaustauschmodell wird durch das CAEX-Schema Version 2.15 beschrieben. Sie können diese Datei auf der Homepage AutomationML e.V. (https://www.automationml.org/o.red.c/dateien.html) herunterladen.

8.5.2 Pruned AML

Einleitung

Pruning beschreibt den Vorgang der Optimierung des Inhalts durch Entfernen bestimmter Dinge, die nicht unbedingt angegeben werden müssen. Bei externen Werkzeugen wie EPLAN haben die automatisch erstellten Submodulinformationen in einer Hardware-Konfiguration keine Bedeutung hinsichtlich EPLAN. Deshalb generieren diese Werkzeuge eine AML-Datei, indem die automatisch erstellten Submodulinformationen aus der Hardware-Konfiguration entfernt werden. Diese Datei wird Pruned AML genannt.

Pruned AML generieren

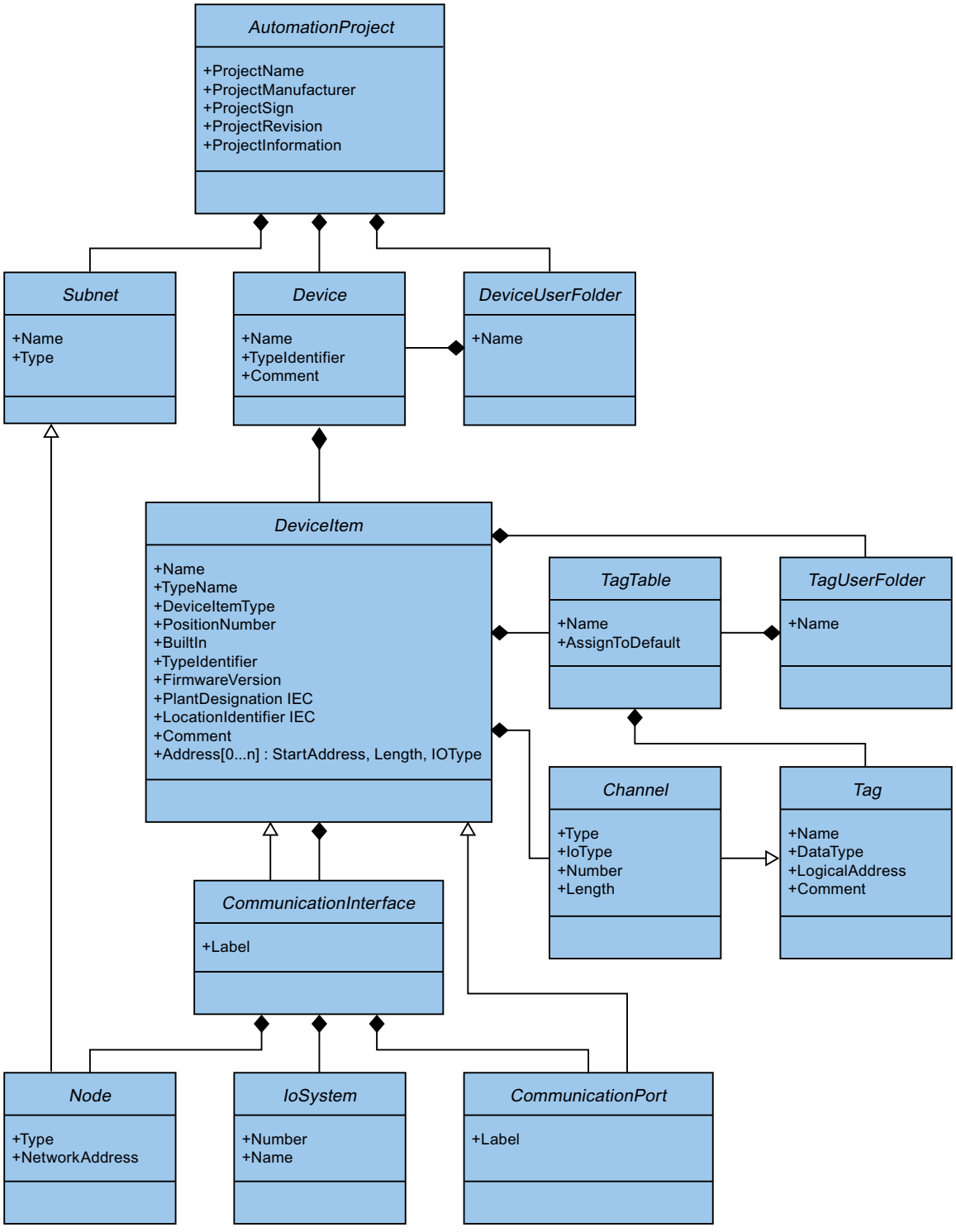
Die Generierung von Pruned AML basiert auf den folgenden Regeln in dieser Reihenfolge.

1. Wenn ein Geräteelement steckbar ist, wird kein Pruning durchgeführt.
2. Wenn ein Geräteelement vom Typ "Schnittstelle" oder "Port" ist, wird kein Pruning durchgeführt.
3. Adressobjekte vom Typ "Diagnose" sind für den Pruning-Algorithmus nicht relevant.
4. Adressobjekte, die mit den automatisch erstellten Submodulen verknüpft sind, werden unter dem direkt übergeordneten Element angegeben (bei dem es sich um ein nicht automatisch erstelltes Submodul handeln muss).
5. Adressobjekte werden in die gleiche Sequenz eingeschlossen, die auch von TIA Portal Openness ausgegeben wird.

8.5.3 Übersicht über Objekte und Parameter von CAx-Import/-Export

Export/Import-Objekte und -Attribute

Die folgende Abbildung zeigt exportierbare Objekte mit ihren Attributen und Abhängigkeiten im CAx-Import/Export.


AutomationProject
+ProjectName
+ProjectManufacturer
+ProjectSign
+ProjectRevision
+ProjectInformation
Subnet
+Name
+Type
Device
+Name
+TypeIdentifier
+Comment
DeviceUserFolder
+Name
DeviceItem
+Name
+TypeName
+DeviceItemType
+PositionNumber
+BuiltIn
+TypeIdentifier
+FirmwareVersion
+PlantDesignation IEC
+LocationIdentifier IEC
+Comment
+Address[0...n] : StartAddress, Length, IOType
TagTable
+Name
+AssignToDefault
TagUserFolder
+Name
Channel
+Type
+IoType
+Number
+Length
Tag
+Name
+DataType
+LogicalAddress
+Comment
CommunicationInterface
+Label
Node
+Type
+NetworkAddress
IoSystem
+Number
+Name
CommunicationPort
+Label

8.5.4 Struktur der CAx-Daten zum Import/Export

Grundstruktur einer Exportdatei

Die Daten in der Exportdatei vom Import/Export sind mit Verweis auf eine Grundstruktur gegliedert. Die Exportdatei wird in einem AML-Format erzeugt.

Die AML-Datei beginnt mit einer Dokumentinformation. Die Datei beinhaltet die Daten der rechnerspezifischen Installation des exportierten Projekts.

Die Exportdatei umfasst die folgenden zwei Abschnitte:

- Weitere Informationen
  `<WriterHeader>` enthält Informationen zum Export- oder Importprozess. Der Import ignoriert den Inhalt des Abschnitts `<AdditionalInformation>`.
  Zum Beispiel können Sie einen Baustein mit `<AdditionalInformation>...</AdditionalInformation>` einfügen und darin zusätzliche Informationen über die Validierung unterbringen. Nach Weiterleitung der AML-Datei kann der Empfänger anhand dieses Bausteins vor dem Import prüfen, ob die AML-Datei geändert wurde.

```
<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>1d4fcebb-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>
  ...
```

- Instanzhierarchie
  Dieser Abschnitt enthält die hierarchische Reihenfolge der exportierten internen Elemente.

```
  ...
    <InstanceHierarchy Name="APC Sample Instance Hierarchy">
    <InternalElement ID="d4dc896a-f4a5-41b6-9c48-8d3a0a5a4343" Name="CAx_asterisk_AML_03_V14">
      <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
      <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
      <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
      <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
      <InternalElement ID="544f3a69-5f65-45ba-ac2f-1448db9493fd" Name="PN/IE_1">
        ...
      </InternalElement>

      <InternalElement ID="12116ac0-94b7-49d2-888d-7d39bbc0caf5" Name="S71500/ET200MP station_1">
        ...
      </InternalElement>
      <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
      <InternalLink Name="Link To Port_1" RefPartnerSideA=
        "726148ce-de5b-4728-8886-4ba273435479:CommunicationPortInterface" RefPartnerSideB=
        "cb9d24f3-8200-4c89-9b40-24ae850e293e:CommunicationPortInterface" />
      <InternalLink Name="Link To Port_2" RefPartnerSideA=
        "cb9d24f3-8200-4c89-9b40-24ae850e293e:CommunicationPortInterface" RefPartnerSideB=
        "726148ce-de5b-4728-8886-4ba273435479:CommunicationPortInterface" />
      .....
      <InternalLink Name="Link To IoSystem_3" RefPartnerSideA=
        "d8f6e006-3778-4a05-aab1-df844fe822fe:LogicalEndPoint_Interface" RefPartnerSideB=
        "2344b7af-329c-4215-92d1-6143b4627b56:LogicalEndPoint_IoSystem" />
    </InternalElement>
  </InstanceHierarchy>
</CAEXFile>
```

Interne Elemente

Alle Objekte innerhalb der Instanzhierarchie der AML-Datei sind `InternalElements`. Das interne Element `AutomationProject` enthält alle internen Elemente aller Rollenklassifizierungen. Jedes interne Element unterstützt ein Set von Attributen.

Das Attribut `<TypeIdentifier>` identifiziert jeden Objekttyp eines Hardware-Objekts, das über TIA Portal Openness erstellt werden kann.

Hinweis

Automatisch erzeugte Objekte

Automatisch erzeugte Objekte können nur von anderen Objekten erzeugt werden. Sie haben keine Attribute und keine Typkennung. Sie werden in die exportierte Datei aufgenommen, doch Sie können den Export eines bestimmten automatisch erstellten Objekts nicht anstoßen.

Am Ende des AML-Elements eines internen Elements wird Folgendes definiert:

- Rollenklassifizierung
  Das Element `SupportedRoleClass` definiert den Objekttyp eines internen Elements. Der Objekttyp wird in der Rollenklassifizierungsbibliothek definiert, die den Standard-AML zu dem Objektmodell von TIA Portal Openness und TIA Portal abbildet.

```
...
<InternalElement ID="1d1a37ed-19d9-4a23-bc91-51f5a8e0244b" Name="Ungrouped devices">
  <InternalElement ID="ab193f5d-0375-4a6d-a576-a903e2b77cca" Name="ET 200SP station_1">
    ...
      <InternalElement ID="72d41729-90a7-4de3-9708-a8eeda6b1886" Name="IO device_1">
        ...
        <SupportedRoleClass RefRoleClassPath="
          AutomationProjectConfigurationRoleClassLib/DeviceItem" />
      </InternalElement>
      ...
      <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/Device" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
  </InternalElement>
  ...
```

- Interne Links
  Das Element `InternalLink` definiert die Kommunikationspartner einer Verbindung.

```
  ...
  <InstanceHierarchy Name="APC Sample Instance Hierarchy">
    <InternalElement ID="d4dc896a-f4a5-41b6-9c48-8d3a0a5a4343" Name="CAx_asterisk_AML_03_V14">
      <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
      <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
      <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
      <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
        ...
      <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
      <InternalLink Name="Link To Port_1" RefPartnerSideA=
        "726148ce-de5b-4728-8886-4ba273435479:CommunicationPortInterface" RefPartnerSideB=
        "cb9d24f3-8200-4c89-9b40-24ae850e293e:CommunicationPortInterface" />
      <InternalLink Name="Link To Port_2" RefPartnerSideA=
        "cb9d24f3-8200-4c89-9b40-24ae850e293e:CommunicationPortInterface" RefPartnerSideB=
        "726148ce-de5b-4728-8886-4ba273435479:CommunicationPortInterface" />
      <InternalLink Name="Link To Port_3" RefPartnerSideA=
        "65307e3e-95fd-41ac-9982-e5e4ffc2fb15:CommunicationPortInterface" RefPartnerSideB=
        "58b1a3f2-f94b-48d1-ab5e-fbc4857cdfbc:CommunicationPortInterface" />
      <InternalLink Name="Link To Port_4" RefPartnerSideA=
        "58b1a3f2-f94b-48d1-ab5e-fbc4857cdfbc:CommunicationPortInterface" RefPartnerSideB=
        "65307e3e-95fd-41ac-9982-e5e4ffc2fb15:CommunicationPortInterface" />
      ...
    </InternalElement>
  </InstanceHierarchy>
</CAEXFile>
```

Attribute

Attribute werden internen Elementen wie folgt zugeordnet:

```
...
<InternalElement ID="1d1a37ed-19d9-4a23-bc91-51f5a8e0244b" Name="Ungrouped devices">
  <InternalElement ID="ab193f5d-0375-4a6d-a576-a903e2b77cca" Name="ET 200SP station_1">
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>System:Device.ET200SP</Value>
    </Attribute>
    <InternalElement ID="7636c362-a7af-47bb-8a18-e6428a6d61ff" Name="Rack_0">
      <Attribute Name="TypeName" AttributeDataType="xs:string">
        <Value>Rack</Value>
      </Attribute>
      <Attribute Name="PositionNumber" AttributeDataType="xs:int">
        <Value>0</Value>
      </Attribute>
      <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
        <Value>False</Value>
      </Attribute>
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>System:Rack.ET200SP</Value>
      </Attribute>
      ...
      <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/Device" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
  <InternalLink Name="Link To Port_1" RefPartnerSideA=
      "5758e2ff-3974-41e9-8bcc-b61a23f1bb58:CommunicationPortInterface"
      RefPartnerSideB="46683602-5129-4504-a9d1-48e6421e2cf0:CommunicationPortInterface" />
  ...
</InternalElement>
```

Handhabung von Attributen

Die Handhabung von Attributen wird für jedes Attribut einzeln wie folgt definiert:

- Ignoriert
  Das Attribut wird beim Import ignoriert und ist in der Exportdatei nicht vorhanden.
- Obligatorisch
  Das Attribut muss in einer Importdatei vorhanden sein und kann in der Exportdatei nicht gelöscht werden.
- Optional
  Falls dieses Attribut in der Importdatei fehlt, wird der Standardwert für das Attribut festgelegt. Dieses Attribut fehlt in der Exportdatei, wenn es für ein Objekt nicht verwendbar ist, z. B. haben nicht alle Module eine Firmware-Version.

- Nur Exportieren
  Das Attribut wird intern durch das TIA Portal bestimmt, z. B. der Typenname des Geräteelements. Wenn dieses in einer Importdatei vorhanden ist, wird es durch das TIA Portal während des Imports ignoriert.
- Nur Importieren
  Das Attribut kann das Importverhalten beeinflussen. Falls dieses Attribut in der Importdatei fehlt, wird das Verhalten dem Standardwert für das Attribut entsprechen.

Siehe auch

AML-Typkennungen (Seite 571)

8.5.5 AML-Typkennungen

Interne Elemente

Der String `TypeIdentifier` besteht aus den folgenden Teilen:

- `<TypeIdentifierType>:<Identifier>`

Die folgenden Werte für `TypeIdentifierType` sind hier zulässig:

- `OrderNumber` dient der Angabe von physischen Modulen
- `GSD` dient der Angabe von GSD/GSDML-basierten Geräten
- `System` dient der Angabe von Systemen und generischen Geräten

Typkennung: OrderNumber

`OrderNumber` ist die allgemeine Typkennung für alle Module im Hardware-Katalog, ausgenommen GSD. AML-Typkennungen entsprechen nicht immer den TIA Portal Openness-Typkennungen. AML-Typkennungen haben keine Information zur `FirmwareVersion`. Die Information zur Firmwareversion ist im getrennten AML-Attribut "`FirmwareVersion`" enthalten.

Das Format für diese `Typkennung` ist eines der folgenden:

- `<OrderNumber>`
  Beispiel: `Bestellnummer: 3RK1 200-0CE00-0AA2`

Hinweis

Platzhalter in den Bestellnummern

Es gibt einige wenige Module im Hardware-Katalog, die „Platzhalter“-Zeichen in ihren Bestellnummern verwenden, die eine bestimmte Gruppe realer Hardware repräsentieren, z. B. die verschiedenen Längen der S7-300-Baugruppenträger.

In diesem Fall können die spezifischen`OrderNumber` und die „Platzhalter“ `OrderNumber` beide verwendet werden, um eine Instanz des Hardware-Objekts zu erstellen. Sie können die Platzhalter jedoch nicht beliebig an jeder Stelle verwenden. Beispiel: Ein S7-300-Baugruppenträger kann auf die folgenden Arten erstellt werden:

`Bestellnummer: 6ES7 390-1***0-0AA0`

oder

`Bestellnummer: 6ES7 390-1AE80-0AA0`

Beachten Sie, dass Sie die folgende Struktur beispielsweise nicht verwenden dürfen:

`Bestellnummer: 6ES7 390-1AE80-0A*0`

Der Rückgabewert der ausgelesenen Typkennung ist immer die Bestellnummer aus dem Hardwarekatalog.

Beispiel: `Bestellnummer: 6ES7 390-1AE80-0AA0` entspricht `Bestellnummer: 6ES7 390-1***0-0AA0`

Typkennung: GSD

Die Typkennung für GSD- und GSDML-basierte Geräte lautet `TypeIdentifier = GSD:<Identifier>`

Die Kennung besteht aus folgenden Elementen

- `GsdName`: Name der GSD oder GSDML in Großbuchstaben
- `GsdType`: Ist eines der folgenden:
  - D: Gerät (Device)
  - R: Baugruppenträger (Rack)
  - DAP: Kopfmodul
  - M: Modul
  - SM: Submodul
- `GsdId`: ID der GSD oder GSDML

Die folgenden Formate für die Typkennungen werden beim CAx-Import/-Export unterstützt:

- `GSD.<GsdName>/<GsdType>`
  Beispiele:
  `GSD:SIEM8139.GSD/DAP`
  `GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCP-20140313.XML/D`
- `<GsdName>/<GsdType>/<GsdId>`
  Beispiel:
  `GSD:SIEM8139.GSD/M/4`
  `GSD:GSDML-V2.31-SIEMENS-SINAMICS_G110M-20140704.XML/M/IDM_DRIVE_47`

Typkennung: System

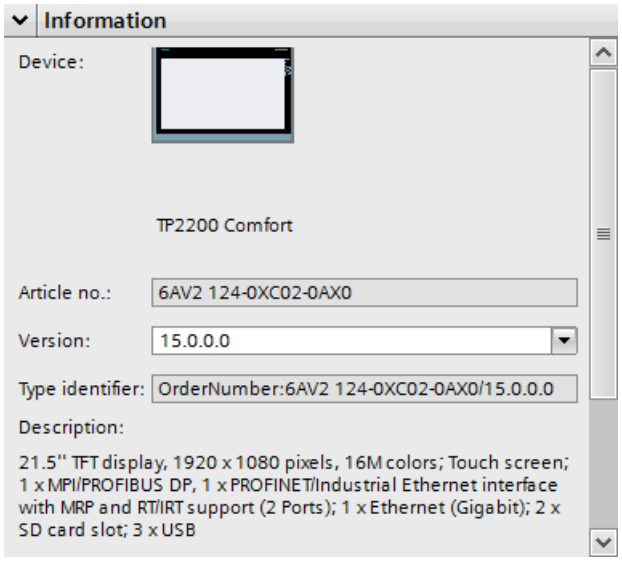
`System.` ist die Kennung für Objekte, die nicht durch andere Kennungen bestimmt werden können. Die Formate für diese `TypeIdentifierType` sind die folgenden:

- `<SystemTypeIdentifier>`
  Beispiele:
  `System:Device.S7300`
  `System:Subnet.Ethernet`
- `<SystemTypeIdentifier>/<AdditionalTypeIdentifier>`
  Die `AdditionalTypeIdentifier` ist erforderlich, wenn `SystemTypeIdentifier` nicht eindeutig ist.
  Die `SystemTypeIdentifier` hat ein Präfix für bestimmte Objekttypen:
  `Subnet.`
  `Device.`
  `Rack.`
  Beispiel: `System:Rack.S71600/Large`
  Ein Baugruppenträger mit Bestellnummer wird über die `OrderNumber` identifiziert.

Typkennungen im TIA Portal anzeigen

Wenn Sie eine Typkennung kennen müssen, ermitteln Sie sie im TIA Portal wie folgt:

1. Aktivieren Sie unter "Optionen > Einstellungen > Hardware-Konfiguration > Anzeige der Typkennung" die Einstellung "Anzeige der Typkennung für Geräte und Module".
2. Öffnen Sie den Editor "Geräte & Netze".
3. Wählen Sie ein Gerät im Katalog aus.
   Die Typkennung wird unter "Information" angezeigt.



8.5.6 Export von CAx-Daten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Im TIA Portal können Sie Ihre Konfiguration im Editor "Geräte & Netze" in eine AML-Datei exportieren. Mit dieser auf TIA Portal Openness basierenden Funktion lassen sich Hardwaredaten von der Projekt- oder Geräteebene exportieren.

TIA Portal Openness bietet die folgenden Möglichkeiten zum Exportieren von CAx-Daten:

- Exportfunktion
  Die Exportfunktion ist über den Dienst `CaxProvider` zugänglich. Für den Dienst `CaxProvider` rufen Sie die Methode `GetService` am Objekt `Project` auf.
- Befehlszeilenschnittstelle
  Sie führen "Siemens.Automation.Cax.AmiHost.exe" unter "C:\Programme\Siemens \Automation\Portal V..\Bin\" aus, indem Sie spezifische Argumente an der Befehlszeile übergeben.

Einschränkungen für CAx in Bezug auf das Exportieren und Importieren

Nicht unterstützt von CAx wird das Exportieren und Importieren von

- Port-Port-Verbindungen
- Verbindungen zu und zwischen Erweiterungsbaugruppenträgern
- Multi-CPUs
- H-Geräten
- HMI-Geräten außer Push Button Panels und Key Panels
- Antrieben
- Ausgabemodus und -bereich von analogen Kanälen
- Gepackten Adressen

Nicht unterstützt von CAx wird das Exportieren und Importieren folgender Geräte und Antriebe:

- 6AV2 104-0XXXX-XXXX
- 6AV2 155-0XXXX-XXXX
- 6ES7 XXX-XXXXX-XXXX
- 6ES7 370-0AA01-0AA0
- 6ES7 451-3AL00-0AE0
- 6GK5 414-3FC00-2AA2
- 6GK5 414-3FC10-2AA2
- 6GK5 495-8BA00-8AA2
- 6GK5 496-4MA00-8AA2
- 6GK5 602-0BA00-2AA3
- 6GK5 602-0BA10-2AA3
- 6GK5 612-0BA00-2AA3
- 6GK5 612-0BA10-2AA3
- 6GK5 613-0BA00-2AA3
- 6GK5 623-0BA10-2AA3
- 6GK5 627-2BA10-2AA3
- System:Device.Scalance/S627

- System:IPIProxy.Device
- System:IPIProxy.Rack

Programmcode: Auf den Dienst `CaxProvider` zugreifen

Ändern Sie den folgenden Programmcode, um auf den Dienst `CaxProvider` zuzugreifen:

```
Project project = tiaPortal.Projects.Open(...);
CaxProvider caxProvider = project.GetService<CaxProvider>();

if(caxProvider != null)

{

      // Perform CAx export and import operation

}
```

CAx-Export auf Projektebene

Um CAx-Daten auf Projektebene zu exportieren, verwenden Sie die Methode `Export` mit den folgenden Parametern:

Name	Beispiel	Beschreibung
ProjectToExport	tiaPortal.Projects[0]	Projektobjekt zum Exportieren
ExportFilePath	new FileInfo(@"D:\Temp\ProjectExport.aml")	Vollständiger Exportdateipfad der AML-Datei
LogFilePath	new FileInfo(@"D:\Temp\ProjectExport_Log.log")	Vollständiger Dateipfad der Protokolldatei

Um CAx-Daten auf Projektebene zu exportieren, ändern Sie folgenden Programmcode:

```
caxProvider.Export(project, new FileInfo(@"D:\Temp\ProjectExport.aml"),
new FileInfo(@"D:\Temp\ProjectExport_Log.log"));
```

CAx-Export auf Geräteebene

Um CAx-Daten auf Geräteebene zu exportieren, verwenden Sie die Methode `Export` mit den folgenden Parametern:

Name	Beispiel	Beschreibung
DeviceToExport	project.Devices[0]	Geräteobjekt zum Exportieren
ExportFilePath	new FileInfo(@"D:\Temp\ProjectExport.aml")	Vollständiger Exportdateipfad der AML-Datei
LogFilePath	new FileInfo(@"D:\Temp\ProjectExport_Log.log")	Vollständiger Dateipfad der Protokolldatei

Um CAx-Daten auf Projektebene zu exportieren, ändern Sie folgenden Programmcode:

```
caxProvider.Export(device, new FileInfo(@"D:\Temp\DeviceExport.aml"), new
FileInfo(@"D:\Temp\DeviceExport_Log.log"));
```

CAx-Export über Befehlszeile

Um CAx-Daten über die Befehlszeile zu exportieren, verwenden Sie Siemens.Automation.Cax.AmiHost.exe mit den folgenden Parametern:

Parameter	Beispiel	Beschreibung
-p	-p "D:\Temp\MyProject.ap14"	Gibt einen vollständigen Pfadnamen zu einem TIA Portal-Projekt an.
-d	-d "S7300/ET200M station_1"	Optionaler Parameter. Sofern kein Gerät angegeben ist, erfolgt der Export auf Projektebene. Gibt den Namen des Geräts oder der Station innerhalb des angegebenen TIA-Projekts an, das exportiert werden soll.
-m	-m "AML"	Gibt den Export-/Importmodus an (Format für Export/Import):. "AML" exportiert in das AML-Format
-e	-e "D:\Import" -e "D:\Import\CAx_Export.aml"	Gibt den vollständigen Pfad der zu exportierenden AML-Datei an. Der Projektname wird als Name der exportierten Datei verwendet, falls nur ein Pfad angegeben ist.

Um CAx-Daten auf Projektebene über die Befehlszeile zu exportieren, ändern Sie folgenden Programmcode:

```
Siemens.Automation.Cax.AmiHost.exe -p "D:\Temp\MyProject.ap14" -m "AML" -e
"D:\Import\CAx_Export.aml"
```

Um CAx-Daten auf Geräteebene über die Befehlszeile zu exportieren, ändern Sie folgenden Programmcode:

```
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 Export/Import von Submodulen

Voraussetzungen

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Der PLC ist offline.

Anwendung

Sie können Daten von Submodulen zwischen dem TIA Portal und anderen Engineering-Tools, z. B. einem CAD-Tool wie EPLAN austauschen, wobei während des Exports und Imports eine gemeinsame Hierarchie der Submodule in der AML-Datei beibehalten werden muss. Die Submodule, wie etwa Busadapter, müssen beispielsweise in TIA Portal eine andere interne Hierarchie aufweisen als in anderen Anwendungen (z. B. CAD-Tools wie EPLAN).

AML-Struktur der Exportdatei

Sie können Submoduldaten aus der TIA Portal-Hierarchie in die AML-Dateihierarchie exportieren.

Das folgende Beispiel zeigt die AML-Dateistruktur, die beim Exportieren von Submodulen erzeugt wird.

```
<?xml version="1.0" encoding="utf-8"?>
 <CAEXFile FileName="Project4.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
 <AdditionalInformation>
   <WriterHeader>
     <WriterName>Totally Integrated Automation Portal</WriterName>
     <WriterID>1d4fcebb-1ad6-4881-b01d-bca335d94a46:V1.0</WriterID>
     <WriterVendor>Siemens AG</WriterVendor>
     <WriterVendorURL>www.siemens.com</WriterVendorURL>
     <WriterVersion>15</WriterVersion>
     <WriterRelease>1500.0100.0.0</WriterRelease>
     <LastWritingDateTime>2018-05-03T11:23:10.3011329Z</LastWritingDateTime>
   </WriterHeader>
 </AdditionalInformation>
 <AdditionalInformation AutomationMLVersion="2.0" />
 <AdditionalInformation DocumentVersions="Recommendations">
 <Document DocumentIdentifier="AR APC" Version="1.0.0" />
 </AdditionalInformation>
 <InstanceHierarchy Name="APC Sample Instance Hierarchy">
   <InternalElement ID="6cd7f80f-e049-4958-ba67-630481805bf0" Name="Project4">
    <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
    <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
    <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
    <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
   <InternalElement ID="b27045c4-9cb3-4b8d-916b-85f8100d1602" Name="Ungrouped devices">
   <InternalElement ID="3f770698-940d-49c2-9f77-06fc458e1340" Name="ET 200SP station_1">
 <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
   <Value>System:Device.ET200SP</Value>
 </Attribute>
   <InternalElement ID="6f52fbab-a221-4d54-9368-84c392ca7fec" Name="Rack_0">
   <Attribute Name="TypeName" AttributeDataType="xs:string">
   <Value>Rack</Value>
...
```

Import von Submodulen

Sie können Submodule aus AML-Dateien importieren, die beim vorhergehenden Export erzeugt wurden.

Hinweis

- Die Hierarchie in der AML-Datei darf die interne Hierarchie in TIA Portal nach dem Import nicht beeinflussen.
- Mit älteren TIAP-Versionen erzeugte AML-Dateien sollten ebenfalls fehlerfrei importiert werden.
- Dieses Verhalten in Bezug auf die Änderung/Umwandlung der Hierarchie gilt sowohl für eingebaute als auch für nicht eingebaute Submodule.

Mehrere Submodule an einer gemeinsamen Schnittstelle

In manchen Fällen sind mehrere Submodule an einer gemeinsamen Schnittstelle vorhanden. Beispiel: IO-Device: IM 155-6 PN/3 HF 6ES7 155-6AU30-0CN0/V4.2. Dieses Modul verfügt über zwei nicht eingebaute Busadapter an der gleichen Schnittstelle. In diesem Fall muss es möglich sein, die Busadapter aus der TIAP-Hierarchie in die erforderliche AML-Dateihierarchie zu exportieren. In diesem Beispiel aus der TIAP-Hierarchie hat die PROFINET-Schnittstelle zwei Busadapter, drei Ports und einen Knoten. Hier gehören Port_1 und Port_2 logisch zu BA 2xRJ45 und Port_3 gehört logisch zu BA 2xRJ45_1, obwohl alle drei Ports an einer Schnittstelle zusammengeführt sind.

Beim Export:

- Nur dem ersten Submodul wird die Original-Schnittstelle mit den verbindungsrelevanten Informationen zugewiesen. Hier wird BA 2xRJ45 die Original-Schnittstelle mit Knoten 'IE1', 'Port_1' und 'Port_2' zugewiesen.
- Die anderen Submodule werden einer 'duplizierten' Schnittstelle mit Ports zugewiesen, die logisch zum Submodul gehören. Hier wird BA 2xRJ45_1 eine 'duplizierte' Schnittstelle und Port_3 zugewiesen.
- Wenn das Kopfmodul mit einem Subnetz/IO-System verbunden ist, wird die zugehörige Verbindungsinformation (wie ExternalInterface-Links) nur als Teil des ersten Submoduls exportiert (ExternalInterface-Link zum Subnetz unter 'Node' und ExternalInterface-Link zum IO-System unter 'Interface').
- Die Verbindungsinformationen zur Topologieverschaltung gehören zum jeweiligen Port.

Beim Import:

- Es ist möglich, mehrere Submodule aus einer AML-Datei zu importieren, die beim vorhergehenden Export erzeugt wurde.

Die AML-Datei, die beim Export für obige Konfiguration erzeugt wird, ist nachfolgend dargestellt:

```
<?xml version="1.0" encoding="utf-8"?>
<CAEXFile FileName="MultipleBA_01.aml" SchemaVersion="2.15"
xsi:noNamespaceSchemaLocation="CAEX_ClassModel_V2.15.xsd">
  <AdditionalInformation>
    <WriterHeader>
      <WriterName>Totally Integrated Automation Portal</WriterName>
      <WriterID>1d4fcebb-1ad6-4881-b01d-bca335d94a46:V1.0</WriterID>
      <WriterVendor>Siemens AG</WriterVendor>
      <WriterVendorURL>www.siemens.com</WriterVendorURL>
      <WriterVersion>15</WriterVersion>
      <WriterRelease>1501.0000.0.0</WriterRelease>
      <LastWritingDateTime>2018-05-17T09:36:46.9230179Z</LastWritingDateTime>
    </WriterHeader>
  </AdditionalInformation>
  <AdditionalInformation AutomationMLVersion="2.0" />
  <AdditionalInformation DocumentVersions="Recommendations">
    <Document DocumentIdentifier="AR APC" Version="1.0.0" />
  </AdditionalInformation>
  <InstanceHierarchy Name="APC Sample Instance Hierarchy">
    <InternalElement ID="e005c094-1b0a-42c4-92a0-67c981508c1a" Name="Project45">
      <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
      <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
      <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
      <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
      <InternalElement ID="2782e61d-8c27-46cb-93ea-6b804157ae60" Name="PN/IE_1">
        <Attribute Name="Type" AttributeDataType="xs:string">
          <Value>Ethernet</Value>
        </Attribute>
        <ExternalInterface ID="2d901881-a2bf-4fe7-915f-b2542b346988"
Name="LogicalEndPoint_Subnet" RefBaseClassPath="CommunicationInterfaceClassLib/
LogicalEndPoint" />
        <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/
Subnet" />
   ...
```

Pruning von AML-Dateien

Das Pruning (Bereinigen) beschreibt den Vorgang der Optimierung des Inhalts durch Entfernen bestimmter Dinge, die nicht unbedingt angegeben werden müssen. Informationen über das Pruning von AML-Dateien enthält Pruning von AML-Dateien (Seite 562).

In manchen Fällen kann die Hierarchie der Submodule in TIA Portal und CAD-Tools (wie EPLAN) auf Grund von Pruning der Submodule voneinander abweichen. In diesen Fällen muss TIA Portal den Import von AML-Dateien sowohl mit als auch ohne Pruning unterstützen.

Hinweis

- Der Export von AML-Dateien ohne Pruning wird in TIA Portal immer unterstützt.
- Der Import von AML-Dateien, ob mit oder ohne Pruning, wird in TIA Portal immer unterstützt.

Siehe auch

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

8.5.8 Import von CAx-Daten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)

Verwendung

Im TIA Portal können Sie Ihre Konfiguration im Editor "Geräte & Netze" aus einer AML-Datei importieren. Mit dieser Funktion lassen sich Hardwaredaten von der Projekt- oder Geräteebene importieren.

TIA Portal Openness bietet die folgenden Möglichkeiten zum Exportieren von CAx-Daten:

- Importfunktion
  Die Importfunktion ist über den Dienst `CaxProvider` zugänglich. Für den Dienst `CaxProvider` rufen Sie die Methode `GetService` am Objekt `Project` auf.
- Befehlszeile
  Sie führen "Siemens.Automation.Cax.AmiHost.exe" unter "C:\Programme\Siemens \Automation\Portal V..\Bin\" aus, indem Sie spezifische Argumente an der Befehlszeile übergeben:

Programmcode: Auf den Dienst `CaxProvider` zugreifen

Ändern Sie folgenden Programmcode:

```
//Access the CaxProvider service
Project project = tiaPortal.Projects.Open(...);
CaxProvider caxProvider = project.GetService<CaxProvider>();

if(caxProvider != null)

{
      // Perform Cax export and import operation
}
```

Cax-Import

Um CAx-Daten in ein TIA Portal-Projekt zu importieren, verwenden Sie die Methode `Import` mit den folgenden Parametern:

Name	Beispiel	Beschreibung
ImportFilePath	new FileInfo(@"D:\Temp\ProjectExport.aml")	Vollständiger Importdateipfad der AML-Datei
LogFilePath	new FileInfo(@"D:\Temp\ProjectExport_Log.log")	Vollständiger Dateipfad der Protokolldatei
ImportOptions	CaxImportOptions.MoveToParkingLot CaxImportOptions.RetainTiaDevice CaxImportOptions.OverwriteTiaDevice	Strategien zur Konfliktlösung beim Import in ein bereits vorhandenes, nicht leeres Projekt.

Um CAx-Daten zu importieren, ändern Sie folgenden Programmcode:

```
caxProvider.Import(new FileInfo(@"D:\Temp\ProjectImport.aml"), new
FileInfo(@"D:\Temp\ProjectImport_Log.log"),
CaxImportOptions.MoveToParkingLot);
```

Die folgenden `CaxImportOptions` sind gegeben:

Importoption	Beschreibung
MoveToParkingLot	Gerät(e) mit Namenskonflikt im Projekt beibehalten und diejenigen aus den CAx-Daten in einen Parkplatzordner importieren
RetainTiaDevice	Gerät(e) mit Namenskonflikt im Projekt beibehalten und diejenigen aus den CAx-Daten nicht importieren
OverwriteTiaDevice	Gerät(e) mit Namenskonflikt im Projekt durch diejenigen aus den CAx-Daten überschreiben

CAx-Import über Befehlszeile

Um CAx-Daten über die Befehlszeile zu importieren, verwenden Sie Siemens.Automation.Cax.AmiHost.exe mit den folgenden Parametern:

Parameter	Beispiel	Beschreibung
-p	-p "D:\Temp\MyProject.ap14"	Gibt einen vollständigen Pfadnamen zu einem TIA Portal-Projekt an.
-m	-m "AML"	Gibt den Export-/Importmodus an (Format für Export/Import):. "AML" exportiert in das AML-Format
-i	-i "D:\Import\CAx_Export.aml"	Gibt den vollständigen Pfad der zu importierenden AML-Datei an.
-c	-c "ParkingLot"	Gibt unterschiedliche Strategien an, wenn es nach den Importoptionen einen Konflikt beim Gerätenamen gibt.

Um CAx-Daten über die Befehlszeile zu importieren, ändern Sie folgenden Programmcode:

```
Siemens.Automation.Cax.AmiHost.exe -p "D:\Temp\MyProject.ap14" -m "AML" -i
"D:\Import\CAx_Export.aml"
```

Folgende Importoptionen stehen zur Verfügung:

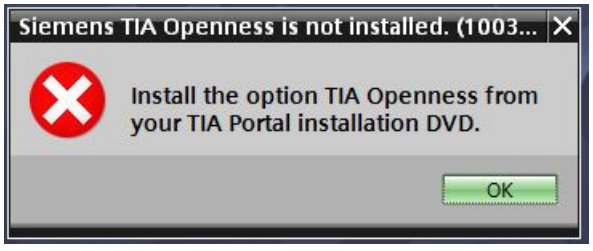
Importoption	Beschreibung
ParkingLot	Gerät(e) mit Namenskonflikt im Projekt beibehalten und diejenigen aus den CAx-Daten in einen Parkplatzordner importieren
RetainTia	Gerät(e) mit Namenskonflikt im Projekt beibehalten und diejenigen aus den CAx-Daten nicht importieren
OverwriteTia	Gerät(e) mit Namenskonflikt im Projekt durch diejenigen aus den CAx-Daten überschreiben

8.5.9 Ausnahmen beim Import und Export von CAx-Daten

Ausnahme aufgrund von Nichtverfügbarkeit von TIA Openness

Die CAx-Implementierung basiert auf TIA Openness Public APIs. Openness Public APIs sind nur verfügbar, wenn der Anwender das Openness-Optionspaket während der Installation des TIA Portals installiert hat. Deshalb muss vor der Ausführung von CAx-bezogenen Funktionen geprüft werden, ob Openness verfügbar ist. (Siehe TIA Openness installieren (Seite 27))

Immer wenn der Anwender eine CAx-Exportaktion oder eine CAx-Importaktion in der Benutzeroberfläche des TIA Portals auslöst, wird eine Prüfung durchgeführt, um zu ermitteln, ob TIA Openness im System vorhanden ist. Wenn keine Installation von TIA Openness gefunden wird, wird dem Anwender ein TIA Portal-Dialog mit der folgenden Fehlermeldung angezeigt.



Wird die CAx-Operation über die Befehlszeile ausgeführt, wird bei Nichtverfügbarkeit von TIA Openness der folgende Fehlerdialog angezeigt.

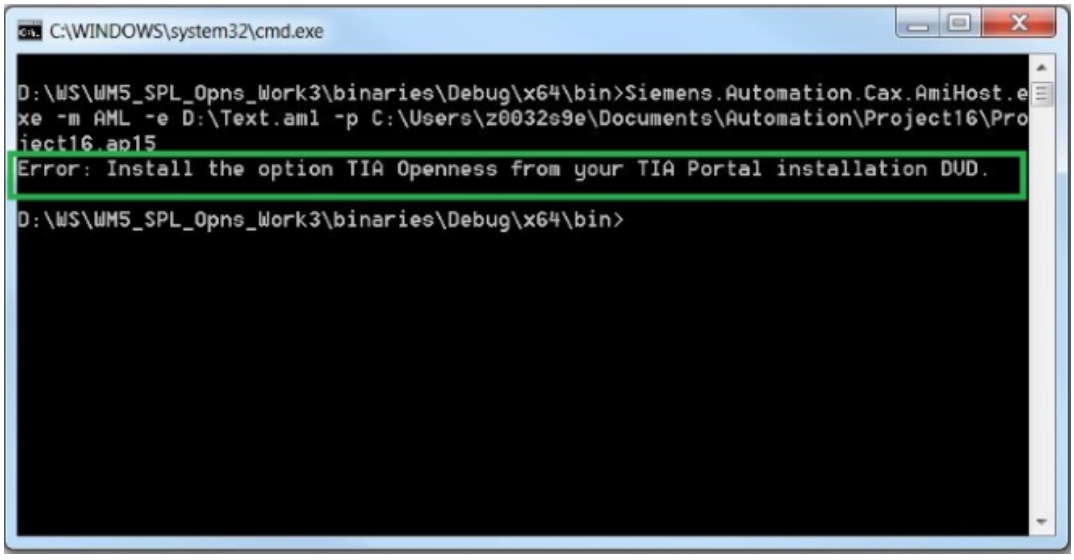


Bild 8-3 OpennessNotInstalled-Commandline

8.5.10 Round-Trip-Geräte und -Module

Voraussetzung

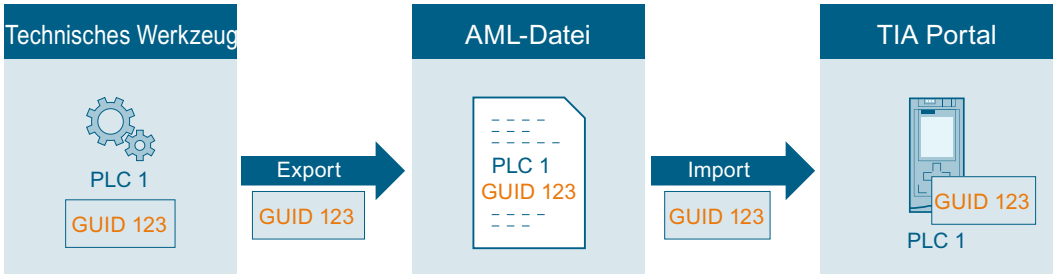
- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist offline.

Verwendung

Sie können zwischen dem TIA Portal und anderen Engineering Tools, beispielsweise einem Elektroplanungs-Tool wie EPLAN oder dem TIA Selection Tool, Konfigurationsdaten austauschen. Zur Identifikation der importierten und exportierten Geräte wird eine eindeutige globale Kennung, die AML GUID, verwendet.

Während der Round-Trips wird die AML GUID für physische Objekte wie Geräte und Geräteelemente, die nicht integriert sind, z. B. CPUs oder Module, stabil gehalten, jedoch nicht für virtuelle Objekte wie Variablen, Kanäle usw.

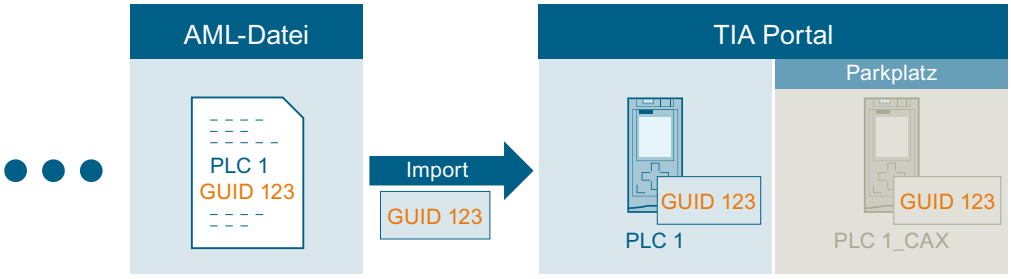
Während des ersten Exports aus dem TIA Portal wird die AML GUID für ein Gerät oder ein nicht integriertes Geräteelement zufällig generiert, danach jedoch stabil gehalten.



Wenn Sie ein Gerät aus einem technischen Werkzeug in ein leeres TIA Portal-Projekt exportieren, wird der AML GUID in den Kommentar des Hardware-Objekts hinzugefügt. Wenn im TIA Portal unter "Werkzeuge > Einstellungen > CAx > Importeinstellungen" die entsprechende Einstellung aktiviert ist, wird die AML GUID in der aktuellen Bearbeitungssprache hinzugefügt. Der Round-Trip-Prozess unterstützt nur eine Editorsprache für die AML GUIDs. Wenn Sie Daten importieren oder exportieren, verwenden Sie stets die Bearbeitungssprache, in der Sie den Round-Trip begonnen haben.

Für alle folgenden Importe und Exporte bleibt der AML GUID für dieses Hardware-Objekt gleich. Änderungen am Hardware-Objekt werden übernommen.

Innerhalb eines TIA Portal-Projekts müssen die Objektnamen eindeutig sein. Der Import eines Geräts oder eines Geräteelements in ein TIA Portal-Projekt, in dem ein Objekt mit dem gleichen Namen bereits vorhanden ist, würde zu einem Namenskonflikt führen. Während des Imports haben Sie die Möglichkeit, die Objekte mit den Namenskonflikten in den benutzerdefinierten Parkplatz zu verschieben. Der Name des importierten Objekts wird um "_CAX" erweitert.



Hinweis

Kopieren eines importierten Geräts

Wenn Sie ein Gerät oder ein Geräteelement mit einer AML GUID kopieren, müssen Sie die AML GUID im Kommentar des kopierten Objekts löschen. Andernfalls existieren Geräte oder Geräteelemente mit identischen AML GUIDs in Ihrem Projekt und führen zu einer ungültigen AML-Datei.

Importeinstellungen

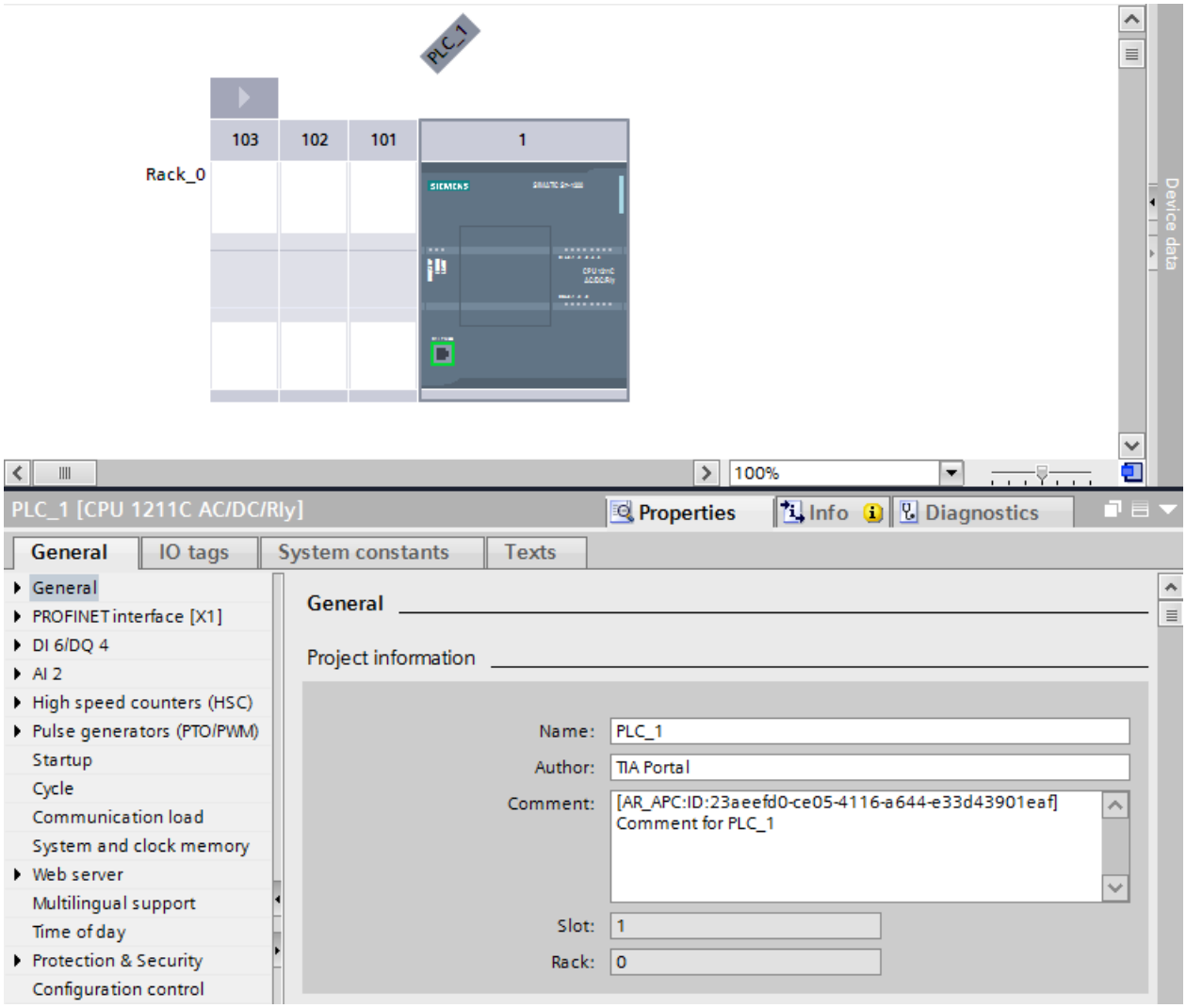
1. Legen Sie den Namen des Parkplatzordners unter "Optionen > Einstellungen > CAx > Einstellungen für Konfliktlösung" fest.
   Der Parkplatzordner dient zum Speichern von Objekten mit Namenskonflikten.
2. Aktivieren Sie "Optionen > Einstellungen > CAx > Importeinstellungen > GUIDs beim Import speichern".

Hinweis

Gültige AML-GUIDs

Wenn Sie einen AML GUID vor dem Import bearbeiten, wird der AML GUID ungültig und der Import wird abgebrochen.

Nach dem Import in das TIA Portal wird der AML GUID wie folgt in den bestehenden Nutzerkommentar hinzugefügt:



Hinweis

Kommentar zu lang

Sollte der Anhang des AML GUID Kommentars die maximale Länge von 500 Zeichen überschreiten, wird der Benutzerkommentar auf 500 Zeichen begrenzt. Eine entsprechende Information wird protokolliert.

AML-Struktur

Die erstellte Kennung wird wie in dem folgenden Code-Ausschnitt beschrieben in eine AML Datei exportiert:

```
<InternalElement ID="23aeefd0-ce05-4116-a644-e33d43901eaf"
Name="PLC_1"
```

8.5.11 Export/Import-Topologie

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist offline.

Verwendung

Im TIA Portal können Sie die Geräte mit ihren Topologieinformationen in eine AML-Datei exportieren. Beim Importieren in ein leeres TIA Portal-Projekt behalten die importierten Geräteelemente die Topologieinformationen bei.

Das Element <InteralLink> liefert Verknüpfungsdetails von Port-zu-Port-Verschaltungen zwischen den Geräteelementen. Es erscheint unter dem gemeinsamen übergeordneten Objekt der verschalteten Geräte und enthält eindeutige Variablennamen.

Attribute eines Elements "InternalLink"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	Die Variablennamen sind als "Link to Port_n" formatiert (dabei steht n für eine Zahl von 1 bis zu der Anzahl von Port-zu-Port-Verschaltungen).
RefPartnerSideA	Obligatorisch	Bezeichnet den verknüpften Port. Formatiert als UniqueIDOfPort:CommunicationPortInterface
RefPartnerSideB	Obligatorisch	Bezeichnet den verknüpften Port. Formatiert als UniqueIDOfPort:CommunicationPortInterface

Beispiel: Topologiesicht

AML-Struktur

Die folgende Abbildung zeigt eine partielle Elementstruktur der exportierten AML-Datei. Sie enthält zwei eindeutige IDs für die Ports in PLCs.

```
...
 <InternalElement ID="e1966b52-b8b3-47b4-8866-a754ebb77648" Name="Port_1">
   <Attribute Name="Label" AttributeDataType="xs:string">


...
 <InternalElement ID="75f31daf-575f-48a2-ab35-8f07a376eb1b" Name="Port_1">
   <Attribute Name="Label" AttributeDataType="xs:string">
```

Das Element <InteralLink> enthält drei obligatorische Attribute.

```
<InternalLink Name="Link to Port_1"
  RefPartnerSideA="e1966b52-b8b3-47b4-8866-a754ebb77648:CommunicationPortInterface"
  RefPartnerSideB="75f31daf-575f-48a2-ab35-8f07a376eb1b:CommunicationPortInterface" />
```

8.5.12 Export eines Geräteelements

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist offline.

Verwendung

Das Objekt `Device` ist ein Behälterobjekt für eine zentrale oder dezentrale Konfiguration. Es ist das übergeordnete Objekt für `DeviceItem`-Objekte und das interne Element oberster Ebene der Instanzhierarchie des TIA Portal-Projekts innerhalb der Struktur einer AML-Datei.

Der CAx-Datenexport unterstützt folgende Arten von Geräten, angegeben über die AML-Typkennung:

- Physische Module
- GSD-/GSDML-basierte Geräte
- Systeme

Geräte können in ein `DeviceUserFolder` Objekt gruppiert werden.

Hinweis

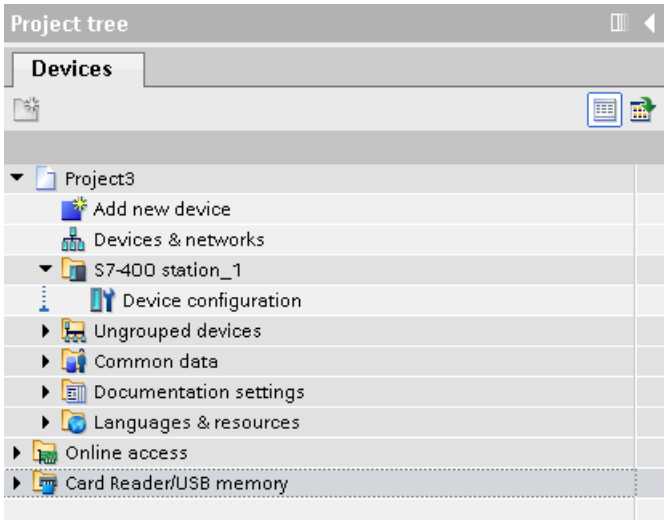
Beim Export eines Einzelgeräts werden auch alle Subnetze des Projekts exportiert.

Attribute

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	
TypeIdentifier	Obligatorisch	
Comment	Optional	Standardeinstellung: „“

Beispiel: Exportierte Konfiguration



AML-Struktur der Exportdatei

Das folgende Strukturbeispiel stellt den Export des Einzelgeräts "S7-400 station_1" ohne Baugruppenträger und Module dar:

```
...
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
  <InternalElement ID="288b7850-688e-43b3-941e-d615ba900a02" Name="Project3">
    <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
    <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
    <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
    <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
    <InternalElement ID="57611cfd-6da4-444e-ac78-5fbcea20a4e1" Name="S7-400 station_1">
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>System:Device.S7400</Value>
      </Attribute>
      <Attribute Name="Comment" AttributeDataType="xs:string">
        <Value>S7400 station</Value>
      </Attribute>
      <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/Device" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
  </InternalElement>
</InstanceHierarchy>
</CAEXFile>
```

Siehe auch

Struktur der CAx-Daten zum Import/Export (Seite 566)

AML-Typkennungen (Seite 571)

8.5.13 Import eines Geräteobjekts

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Der PLC ist offline.

Verwendung

Das Objekt `Device` ist ein Behälterobjekt für eine zentrale oder dezentrale Konfiguration. Es ist das übergeordnete Objekt für `DeviceItem`-Objekte und das interne Element oberster Ebene der Instanzhierarchie des TIA Portal-Projekts innerhalb der Struktur einer AML-Datei.

Der CAx-Datenimport unterstützt folgende Arten von Geräten, angegeben über die AML-Typkennung:

- Physische Module
- GSD-/GSDML-basierte Geräte
- Systeme
- Generische Geräte

Wenn ein `DeviceUserFolder` Objekt bereits in einem TIA Portal-Projekt existiert, werden die Geräte in dem entsprechenden Ordner gruppiert.

Wenn Sie nur die Kennung (`TypeIdentifier`) eines Kopfmoduls oder eines PLC und nicht die des entsprechenden Baugruppenträgers und Geräts kennen, können Sie einen generischen Baugruppenträger importieren.

Beispiel: `TypeIdentifier = System:<Prefix>.Generic`

Für den Austausch generischer Geräte müssen die folgenden Elemente in dem in der AML Datei beschriebenen Baugruppenträger vorhanden sein.

- Zentrale Geräte: SPS
- Dezentrale Geräte: Kopfmodul

Sind die Geräte generisch, definiert das Attribut `BuiltIn` die Art des Baugruppenträgers oder Moduls:

- Physisch: `BuiltIn = True`
- Generisch: `BuiltIn = False`

Beispiel: Generische Geräte importieren

Das folgende Strukturbeispiel stellt den Import des generischen "S7-400 station" -Geräts ohne Baugruppenträger und Module dar.

```
...
  <InstanceHierarchy Name="APC Sample Instance Hierarchy">
  <InternalElement ID="d4dc896a-f4a5-41b6-9c48-8d3a0a5a4343" Name="MyProject">
    <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
    <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
    <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
    <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
    <InternalElement ID="3e6277d1-1b12-4c18-b00e-25e3eac3ac35" Name="S7400 station_1">
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>System:Device.Generic</Value>
      </Attribute>
      <Attribute Name="Comment" AttributeDataType="xs:string">
        <Value>S7400 station_1</Value>
      </Attribute>
      <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/Device" />
    </InternalElement>
     ...
    <SupportedRoleClass RefRoleClassPath=
       "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
  </InternalElement>
 </InstanceHierarchy>
</CAEXFile>
```

Beispiel: Importieren der Hierarchie eines Anwenderordners eines Geräts

Das folgende Strukturbeispiel zeigt den Import einer Ordnerhierarchie.

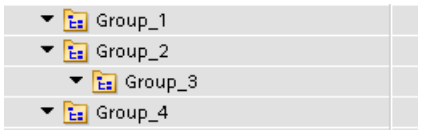
```
...
<InternalElement ID="4fe37f4f-2661-4492-95f0-3d8a8160c851" Name="Project1">
      <Attribute Name="ProjectManufacturer" AttributeDataType="xs:string" />
      <Attribute Name="ProjectSign" AttributeDataType="xs:string" />
      <Attribute Name="ProjectRevision" AttributeDataType="xs:string" />
      <Attribute Name="ProjectInformation" AttributeDataType="xs:string" />
      <InternalElement ID="1ee1615f-9c67-432d-a7cc-b795babf67b6" Name="Group_1">
        <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
      </InternalElement>
      <InternalElement ID="ce14c85a-28de-41aa-ad08-2eb7d0fb755f" Name="Group_2">
        <InternalElement ID="852347e8-3c48-4eb9-8bd8-349d0c7caf34" Name="Group_3">
          <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
        </InternalElement>
        <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
      </InternalElement>
      <InternalElement ID="97cf7924-1756-4e32-8716-ac18990e4762" Name="Group_4">
        <SupportedRoleClass RefRoleClassPath=
        "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
      </InternalElement>
      <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
    </InternalElement>
...
```

Importierte Anwenderordnerhierarchie

Der Name der Ordner für nicht gruppierte und nicht zugeordnete Geräte wird sprachspezifisch vergeben. Deshalb wird empfohlen, einen Import mit derselben Benutzeroberflächensprache wie beim Export durchzuführen. Andernfalls werden nicht gruppierte und nicht zugeordnete Geräte in Ordner importiert, die entsprechend der Sprache des Exports benannt sind.

Beispiel: Sie haben ein Projekt exportiert, das die Gerätesystemgruppe "Ungrouped devices" (englisch) enthält, und importieren die AML-Datei in einer deutschen Benutzeroberfläche. Das Ergebnis: Im Projekt wird eine Gerätesystemgruppe "Nicht gruppierte Geräte" (deutsch) angelegt, gleichzeitig wird beim CAx-Import eine Benutzergruppe "Ungrouped device" (englisch) erstellt.

Die folgende Hierarchie wird in die Projektnavigation importiert:



Siehe auch

Struktur der CAx-Daten zum Import/Export (Seite 566)

AML-Typkennungen (Seite 571)

8.5.14 Export/Import eines Geräts mit festgelegter Adresse

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist offline.

Verwendung

Im TIA Portal können Sie die Adressobjekte von IO-Geräteelementen in eine AML-Datei exportieren. Beim Importieren in ein leeres TIA Portal-Projekt behalten die importierten Geräteelemente die Adressobjekte in den entsprechenden IO-Geräteelementen bei.

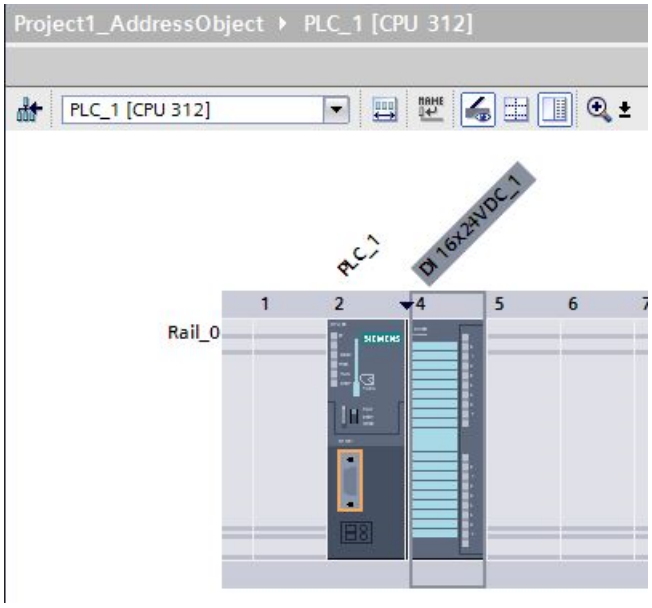
Das Attribut Address in der AML-Datei enthält RefSemantic mit der obligatorischen Einstellung auf den angegebenen Wert OrderedListType.

Attribute eines Elements "Address"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
IoType	Obligatorisch	Eingang oder Ausgang
Length	Optional	Kanalbreite
StartAddress	Obligatorisch	Anfangsadresse des IO-Geräts

Beispiel: IO-Geräteelemente mit Adressobjekten



AML-Struktur

Die folgende Abbildung zeigt eine partielle Elementstruktur der exportierten AML-Datei. Sie enthält die Address-Elemente und deren Attribute.

```
<Attribute Name="Address">
    <RefSemantic CorrespondingAttributePath="OrderedListType" />
    <Attribute Name="1">
      <Attribute Name="StartAddress" AttributeDataType="xs:int">
        <Value>0</Value>
      </Attribute>
      <Attribute Name="Length" AttributeDataType="xs:int">
        <Value>16</Value>
      </Attribute>
      <Attribute Name="IoType" AttributeDataType="xs:string">
        <Value>Input</Value>
      </Attribute>
    </Attribute>
  </Attribute>
```

Pruned XML

Pruning beschreibt den Vorgang der Optimierung des Inhalts durch Entfernen bestimmter Dinge, die in der XML nicht unbedingt angegeben werden müssen. In dieser reduzierten XML werden die automatisch erstellten Submodulinformationen nicht angegeben, und die entsprechenden Adressobjekte werden unter dem direkt übergeordneten Element angeordnet.

Die folgende Abbildung zeigt eine partielle Elementstruktur der exportierten AML-Datei vor dem Pruning.

```
<InternalElement ID="5511a117-42c6-44b7-be5d-0f33cd46e932" Name="AS-i Master_1">
  <Attribute Name="PositionNumber" AttributeDataType="xs:int">
    <Value>4</Value>
  </Attribute>
  <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
    <Value>True</Value>
  </Attribute>
  <Attribute Name="Address">
    <RefSemantic CorrespondingAttributePath="OrderedListType" />
    <Attribute Name="1">
      <Attribute Name="StartAddress" AttributeDataType="xs:int">
        <Value>20</Value>
      </Attribute>
      <Attribute Name="Length" AttributeDataType="xs:int">
        <Value>256</Value>
      </Attribute>
      <Attribute Name="IoType" AttributeDataType="xs:string">
        <Value>Input</Value>
      </Attribute>
    </Attribute>
```

In der Pruned-AML-Datei werden die Submodulinformationen like <InternalElement> entfernt und die entsprechenden Adressobjekte beibehalten.

```
<Attribute Name="Address">
  <RefSemantic CorrespondingAttributePath="OrderedListType" />
  <Attribute Name="1">
    <Attribute Name="StartAddress" AttributeDataType="xs:int">
      <Value>20</Value>
    </Attribute>
    <Attribute Name="Length" AttributeDataType="xs:int">
      <Value>256</Value>
    </Attribute>
    <Attribute Name="IoType" AttributeDataType="xs:string">
      <Value>Input</Value>
    </Attribute>
  </Attribute>
</Attribute>
```

Siehe auch

Pruned AML (Seite 562)

8.5.15 Export/Import eines Geräts mit Kanälen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist offline.

Verwendung

Im TIA Portal können Sie die Kanalobjekte von IO-Geräteelementen in eine AML-Datei exportieren. Beim Importieren in ein leeres TIA Portal-Projekt behalten die importierten Geräteelemente die Kanalobjekte in den entsprechenden IO-Geräteelementen bei.

Das Element <ExternalInterface> wird in internen Elementen von Knoten und Subnetzen dargestellt und gibt an, dass Knoten und Subnetze verbunden sind.

Attribute eines Elements "ExternalInterface"

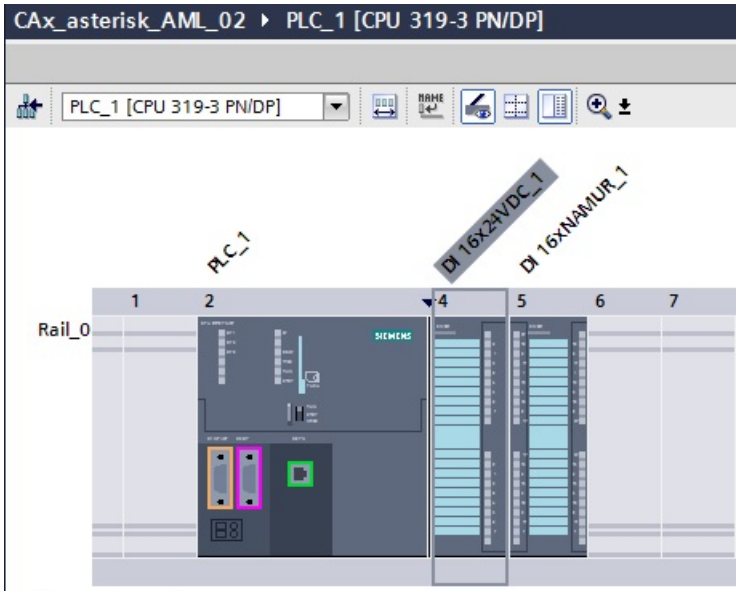
Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
IoType	Obligatorisch	Eingang oder Ausgang
Length	Optional	Kanalbreite (1 für digitale und 16 für analoge Signale)
Number	Obligatorisch	Kanalnummer beginnt bei 0
Type	Obligatorisch	Analog oder digital

Kanalnummerierung

Digitaleingangs-, Digitalausgangs-, Analogeingangs-, Analogausgangs- und Technologiekanäle werden als DI_0, DO_0, AI_0, AO_0,TO_0 nummeriert. Die Kanäle auf den Geräteelementen selbst werden zuerst nummeriert, und die Kanäle auf Untergeräteelementen werden nachfolgend nummeriert (Tiefe zuerst). Jedes weitere Geräteelement hat eine eigene Kanalnummer, beginnend bei 0.

Beispiel: Geräte mit Kanälen



AML-Struktur

Die folgende Abbildung zeigt eine partielle Elementstruktur der exportierten AML-Datei.

```
<ExternalInterface ID="31ca16d3-6322-43b6-95bc-e2d7d7bfc7b7" Name="Channel_DI_0"
                RefBaseClassPath="AutomationProjectConfigurationInterfaceClassLib/Channel">
          <Attribute Name="Type" AttributeDataType="xs:string">
            <Value>Digital</Value>
          </Attribute>
          <Attribute Name="IoType" AttributeDataType="xs:string">
            <Value>Input</Value>
          </Attribute>
          <Attribute Name="Number" AttributeDataType="xs:int">
            <Value>0</Value>
          </Attribute>
          <Attribute Name="Length" AttributeDataType="xs:int">
            <Value>1</Value>
          </Attribute>
        </ExternalInterface>
```

8.5.16 Export von Geräteelementobjekten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist offline.

Verwendung

Der Export von Geräteelementobjekten ist nur für PLC-Geräte anwendbar.

`DeviceItem`-Objekte sind verschachtelte untergeordnete Elemente eines `Device`-Objekts: Ein Objekt vom Typ `DeviceItem` kann ein Baugruppenträger oder ein Einschubmodul sein.

- Das erste Nachfolgeelement eines Geräts ist vom Typ „Baugruppenträger“. Die `PositionNumber` eines Baugruppenträgers beginnt bei 0. Gibt es mehrere Baugruppenträger, werden diese fortlaufend nummeriert (1, 2, 3, ...).
  Es gibt keine Einschränkungen bezüglich der Reihenfolge innerhalb eines Hierarchielevels in der AML Datei.
- Alle weiteren Nachfolger des Typs „Baugruppenträger“ sind Module.

Der CAx-Datenexport unterstützt folgende Arten von Geräteelemente, angegeben über die AML-Typkennung:

- Physische Module
- GSD/GSDML-Module

Attribute

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

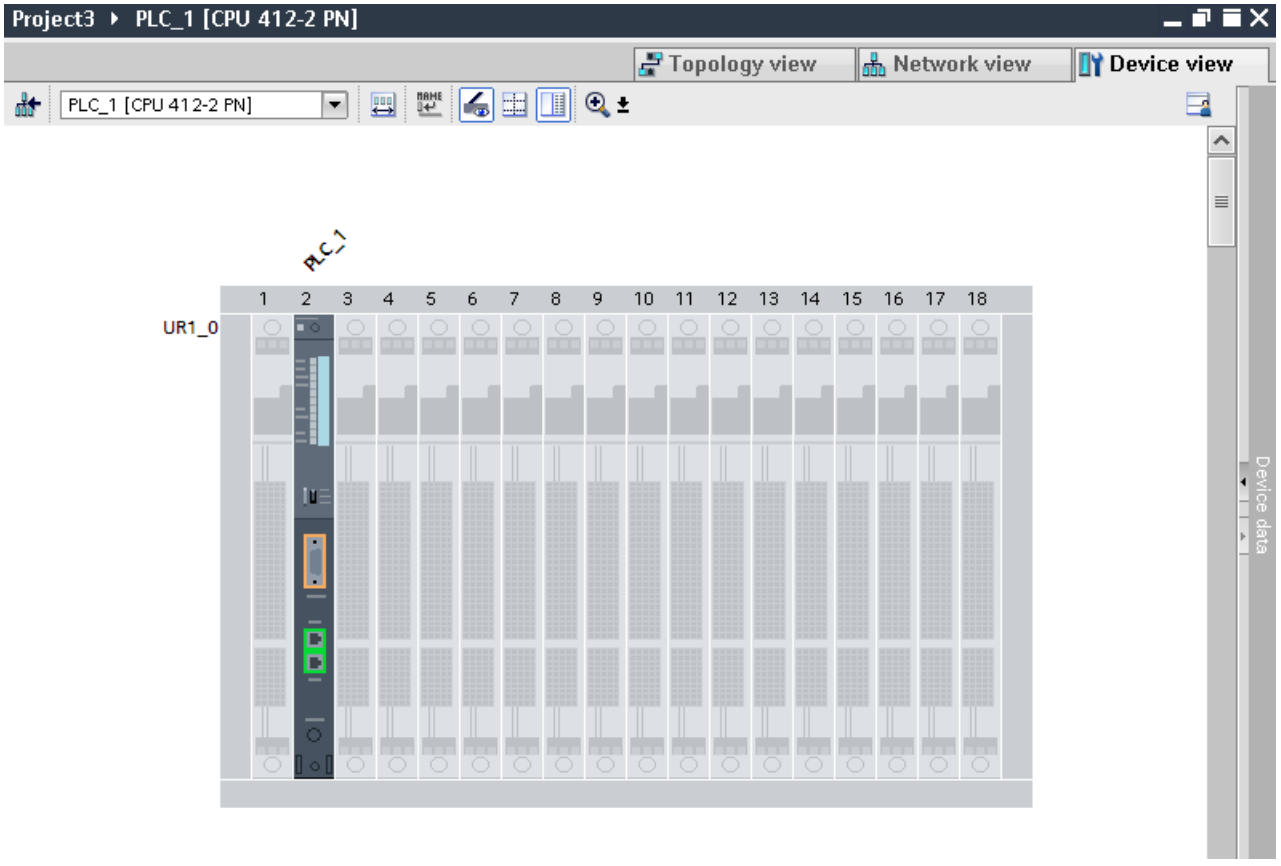
Attribut	Handhabung	Kommentar
Name	Obligatorisch Nur Exportieren für „BuiltIn“ = TRUE	
TypeName	Nur Exportieren für „BuiltIn“ = FALSE	
DeviceItemType	Nur Exportieren	Nur für PLCs (Zentralgeräte) und Geräteelemente (physische Baugruppenträger, Module, Kopfmodul).
PositionNumber	Obligatorisch	
BuiltIn	Optional	Standardeinstellung: FALSE
TypeIdentifier	Obligatorisch für „BuiltIn“ = FALSE Ignoriert für „BuiltIn“ = TRUE	

Attribut	Handhabung	Kommentar
FirmwareVersion	Optional, obligatorisch wenn das Objekt Firmware-Versionen unterstützt	
PlantDesignation IEC	Optional	Standardeinstellung: „“
LocationIdentifier IEC	Optional	Standardeinstellung: „“
Comment	Optional für „BuiltIn“ = FALSE	Standardeinstellung: „“
Address	Optional	"Address" besitzt verschachtelte Attribute

Die folgende Tabelle zeigt die verschachtelten Attribute der „Address“ Attribute des Objekts „DeviceItem“:

Attribute für „Address“	Handhabung	Kommentar
StartAddress	Obligatorisch	
Length	Nur Exportieren	Der Export/Import von Adressen mit der Länge = 0 wird nicht unterstützt.
IoType	Obligatorisch	Eingang oder Ausgang

Beispiel: Exportierte Konfiguration



AML-Struktur der Exportdatei

Das folgende Strukturbeispiel stellt den Export von „UR1_0“ und dem Modul „PLC_1“ dar.

```
...
<InternalElement ID="6563466e-2de9-42ca-951d-eb8f2545958d" Name="S7-400 station_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Device.S7400</Value>
  </Attribute>
  <InternalElement ID="96930368-14ec-43e2-b9b7-c1fefc4b0534" Name="UR1_0">
    <Attribute Name="TypeName" AttributeDataType="xs:string">
      <Value>UR1</Value>
    </Attribute>
    <Attribute Name="PositionNumber" AttributeDataType="xs:int">
      <Value>0</Value>
    </Attribute>
    <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
      <Value>False</Value>
    </Attribute>
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>OrderNumber:6ES7 400-1TA01-0AA0</Value>
    </Attribute>
    <InternalElement ID="a1de449e-4f89-45af-8bbc-f77c28bccd04" Name="PLC_1">
      <Attribute Name="TypeName" AttributeDataType="xs:string">
        <Value>CPU 412-2 PN</Value>
      </Attribute>
      <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
        <Value>CPU</Value>
      </Attribute>
      <Attribute Name="PositionNumber" AttributeDataType="xs:int">
        <Value>2</Value>
      </Attribute>
      <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
        <Value>False</Value>
      </Attribute>
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>OrderNumber:6ES7 412-2EK06-0AB0</Value>
      </Attribute>
      <Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
        <Value>V6.0</Value>
      </Attribute>
     ...
      <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/Device" />
</InternalElement>
...
```

Siehe auch

Export/Import von GSD/GSDML-basierten Geräten und Geräteelementen (Seite 607)

Struktur der CAx-Daten zum Import/Export (Seite 566)

AML-Typkennungen (Seite 571)

8.5.17 Import von Geräteelementobjekten

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal aufbauen (Seite 74)
- Ein Projekt ist geöffnet.
  Siehe Projekt öffnen (Seite 99)
- Die PLC ist offline.

Verwendung

Der Import von Geräteelementobjekten ist nur für PLC-Geräte anwendbar.

`DeviceItem`-Objekte sind verschachtelte untergeordnete Elemente eines `Device`-Objekts: Ein Objekt vom Typ `DeviceItem` kann ein Baugruppenträger oder ein Einschubmodul sein.

- Das erste Nachfolgeelement eines Geräts ist vom Typ Baugruppenträger. Die `PositionNumber` eines Baugruppenträgers beginnt bei 0. Gibt es mehrere Baugruppenträger, werden diese fortlaufend nummeriert (1, 2, 3, ...).
  Es gibt keine Einschränkungen bezüglich der Reihenfolge innerhalb eines Hierarchielevels in der AML Datei.
- Alle weiteren Nachfolger des Typs Baugruppenträger sind Module.

Der CAx-Datenimport unterstützt folgende Arten von Geräteelemente, angegeben über die AML-Typkennung:

- Physische Module
- GSD/GSDML-Module
- Generische Module

Wenn Sie nur die Kennung (`TypeIdentifier`) eines Kopfmoduls oder eines PLC und nicht die des entsprechenden Baugruppenträgers und Geräts kennen, können Sie einen generischen Baugruppenträger importieren.

Beispiel: `TypeIdentifier` = `System:Rack.Generic`

Für den Austausch generischer Baugruppenträger müssen die folgenden Elemente in dem in der AML Datei beschriebenen Baugruppenträger vorhanden sein:

- Zentrale Geräte: SPS
- Dezentrale Geräte: Kopfmodul

Ein generischer Baugruppenträger stammt aus dem Device-Typ. Demnach kann eine AML-Datei, die in das TIA Portal importiert wird, die Typkennung dieses Baugruppenträgers verwenden:

In diesem Fall bestimmt das TIA Portal die Typkennung für den Baugruppenträger.

Sind Baugruppenträger und Module generisch, definiert das Attribut `BuiltIn` die Art des Baugruppenträgers oder Moduls:

- Physisch: `BuiltIn = True`
- Generisch: `BuiltIn = False`

Einschränkungen

Während des Imports ist das Attribut `DeviceItemType` nicht relevant und ist optional.

Hinweis

Attribut "FirmwareVersion"

Wenn in der Importdatei keine `FirmwareVersion` angegeben ist, wird beim CAx-Import die letzte Firmwareversion verwendet, die im TIA Portal verfügbar ist.

Wenn das Attribut `FirmwareVersion` in der Importdatei mit einem leeren Wert vorhanden ist, schlägt der Import des Geräteelements fehl und es wird eine Fehlermeldung protokolliert.

Beispiel: Generische Geräte importieren

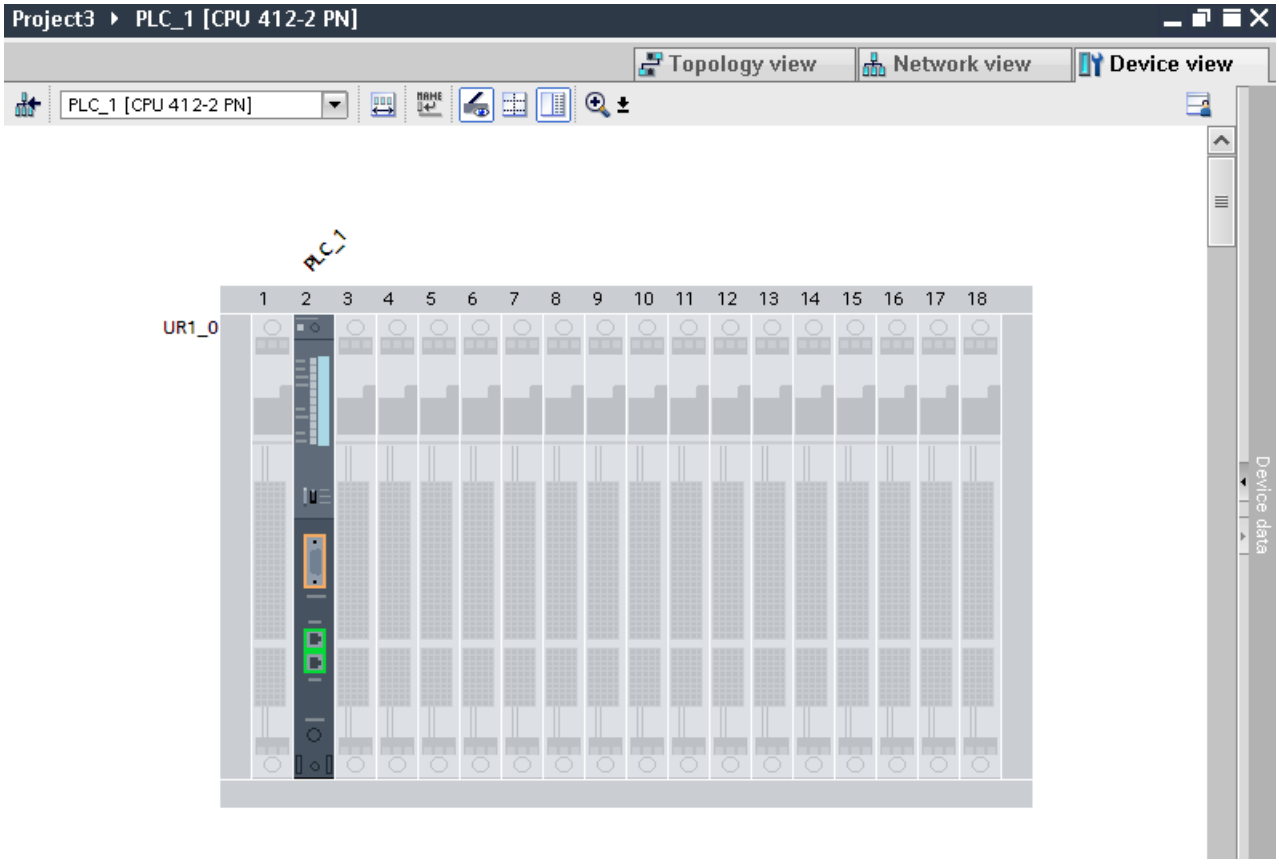
Das folgende Strukturbeispiel stellt den Import des generischen Baugruppenträgers „Rack_1“ dar.

```
...
<InternalElement ID="6563466e-2de9-42ca-951d-eb8f2545958d" Name="S7-400 station_1">
  <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
    <Value>System:Device.S7400</Value>
  </Attribute>
  <InternalElement ID="96930368-14ec-43e2-b9b7-c1fefc4b0534" Name="UR1_0">
    <Attribute Name="TypeName" AttributeDataType="xs:string">
      <Value>UR1</Value>
    </Attribute>
    <Attribute Name="PositionNumber" AttributeDataType="xs:int">
      <Value>0</Value>
    </Attribute>
    <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
      <Value>False</Value>
    </Attribute>
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>System:Rack.Generic</Value>
    </Attribute>
    <InternalElement ID="a1de449e-4f89-45af-8bbc-f77c28bccd04" Name="PLC_1">
      <Attribute Name="TypeName" AttributeDataType="xs:string">
        <Value>CPU 412-2 PN</Value>
      </Attribute>
      <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
        <Value>CPU</Value>
      </Attribute>
      <Attribute Name="PositionNumber" AttributeDataType="xs:int">
        <Value>2</Value>
      </Attribute>
      <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
        <Value>False</Value>
      </Attribute>
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>OrderNumber:6ES7 412-2EK06-0AB0</Value>
      </Attribute>
      <Attribute Name="FirmwareVersion" AttributeDataType="xs:string">
        <Value>V6.0</Value>
      </Attribute>

      <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/Device" />
</InternalElement>
...
```

Importierte Konfiguration

Die folgende Abbildung zeigt die importiere Konfiguration in der TIA Portal-Nutzerschnittstelle:



Siehe auch

Struktur der CAx-Daten zum Import/Export (Seite 566)

AML-Typkennungen (Seite 571)

8.5.18 Export/Import von GSD/GSDML-basierten Geräten und Geräteelementen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts
- Die PLC ist offline.

Verwendung

Der CAx-Import/Export von GSD/GSDML-basierten Geräten und Geräteelementen ist mit dem Import/Export von Standardgeräten vergleichbar.

Bei GSD/GSDML-basierten Geräten und Geräteelementen weichen die exportierbaren Attribute voneinander ab, z. B. bei GSD/GSDML ist das Attribut "`Label`" vorhanden.

Generischer Import von Geräten und Baugruppenträgern ist möglich. Verwenden Sie für den Import die gleiche Kennung wie für Standardgeräte:

- Generische Geräte importieren: `TypeIdentifier = System:Device.Generic`
- Generische Baugruppenträger importieren: `TypeIdentifier = System:Rack.Generic`

Sind die Geräte generisch, definiert das Attribut `BuiltIn` die Art:

- Physisch: `BuiltIn = True`
- Generisch: `BuiltIn = False`

Attribute für ein Gerät

Die folgende Tabelle zeigt die zugehörigen Attribute von Geräten von CAx-Import- und -Exportdateien:

Attribut	Handhabung von Attribut	Kommentar
Name	Obligatorisch für Export und Import	
TypeIdentifier	Obligatorisch für Export und Import	
Comment	Optional für Import	

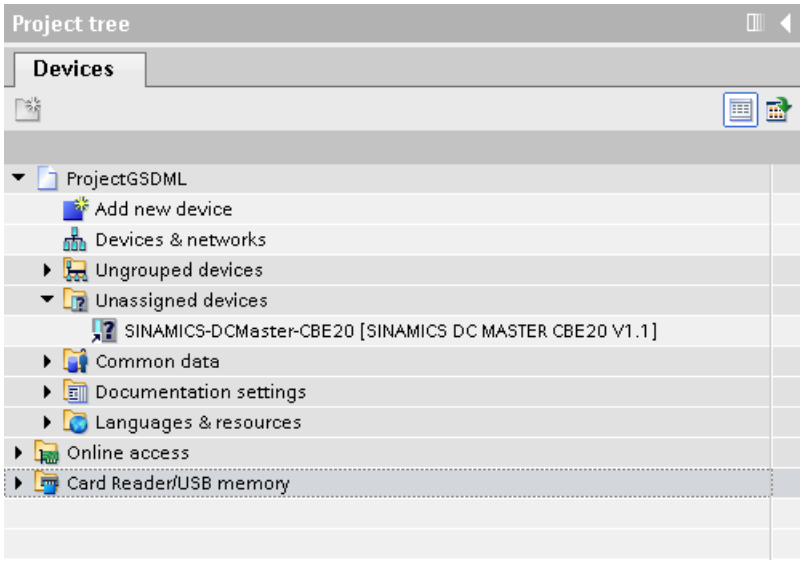
Attribute für ein Geräteelement

Die folgende Tabelle zeigt die zugehörigen Attribute eines Geräteelements von CAx-Import- und -Exportdateien:

Attribut	Handhabung von Attribut Integriert = FALSCH Generische Geräteelemente	Handhabung von Attribut Integriert = WAHR Physische Geräteelemente	Kommentar
Name	Obligatorisch	Nur Exportieren	
TypeName	Nur Exportieren	-/-	
DeviceItem-Type	Nur Exportieren	Nur Exportieren	Nur für SPS- (zentrale Geräte) und Kopfmodul- (dezentrale Geräte) Geräteelemente

Attribut	Handhabung von Attribut Integriert = FALSCH Generische Geräteelemente	Handhabung von Attribut Integriert = WAHR Physische Geräteelemente	Kommentar
PositionNumber	Obligatorisch	Obligatorisch für Export Ausnahmefälle: Geräteelemente vom Typ Schnittstelle: Optional für Import Geräteelemente vom Typ Port: Obligatorisch für Import von integrierten Geräten, wenn Attribut „Label" nicht spezifiziert ist. Wenn „PositionNumber“ und „Label“ beide konfiguriert sind, bekommt „PositionNumber“ die höhere Priorität für Export und Import.	
BuiltIn	Optional		Standardeinstellung: FALSE
TypeIdentifier	Obligatorisch für „BuiltIn“ = FALSE	Ignoriert für „BuiltIn“ = TRUE	
Comment	Optional	-/-	
Label	-	- Geräteelemente vom Typ Schnittstelle: Obligatorisch Geräteelemente vom Typ Port: Obligatorisch wenn Attribut „Positionsnummer“ nicht spezifiziert ist. Wenn „Positionsnummer“ und „Beschriftung“ beide konfiguriert sind, bekommt „Positionsnummer“ die höhere Priorität; das gleiche gilt für den Import.	

Beispiel: Exportiertes GSD-/GSDML-Gerät



AML-Struktur der Exportdatei

Die folgende Abbildung zeigt die Struktur der exportierten AML-Datei.

```
...
<InternalElement ID="9ae02cde-dfb4-4d43-a649-68b9ede7fc3d" Name="Ungrouped devices">
  <InternalElement ID="12d4ce0f-346d-4bfa-b139-c9d0db64c794" Name="GSD device_1">
    <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
      <Value>GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCMASTER-20140704.XML/D</Value>
    </Attribute>
    <InternalElement ID="ccb1cb62-67b2-4b8c-951f-10c7ffb4d787" Name="Rack">
      <Attribute Name="TypeName" AttributeDataType="xs:string">
        <Value>Rack</Value>
      </Attribute>
      ...
      <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
        <Value>GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCMASTER-20140704.XML/R/IDD_14</Value>
      </Attribute>
      <InternalElement ID="74f25b5c-0c09-46d0-9011-f341a3e98a0d" Name="SINAMICS-DCMaster-CBE20">
        <Attribute Name="TypeName" AttributeDataType="xs:string">
          <Value>SINAMICS DC MASTER CBE20 V1.1</Value>
        </Attribute>
        <Attribute Name="DeviceItemType" AttributeDataType="xs:string">
          <Value>HeadModule</Value>
        </Attribute>
        <Attribute Name="PositionNumber" AttributeDataType="xs:int">
          <Value>0</Value>
        </Attribute>
        <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
          <Value>False</Value>
        </Attribute>
        <Attribute Name="TypeIdentifier" AttributeDataType="xs:string">
          <Value>GSD:GSDML-V2.31-SIEMENS-SINAMICS_DCMASTER-20140704.XML/DAP/IDD_14</Value>
        </Attribute>
        <InternalElement ID="94f34bb9-fe47-4904-b8a1-62e8fb6b1b74"
            Name="SINAMICS DC MASTER CBE20 V1.1">
          <Attribute Name="PositionNumber" AttributeDataType="xs:int">
            <Value>0</Value>
          </Attribute>
          <Attribute Name="BuiltIn" AttributeDataType="xs:boolean">
            <Value>True</Value>
          </Attribute>
          <SupportedRoleClass RefRoleClassPath=
              "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
        </InternalElement>
        ...
        <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
      </InternalElement>
      <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/Device" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceUserFolder" />
</InternalElement>
...
```

Siehe auch

AML-Typkennungen (Seite 571)

8.5.19 Exportieren/Importieren von Subnetzen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts
- Die PLC ist offline.

AML-Struktur

Subnetze beschreiben ein physisches Netzwerk, insbesondere, welche Geräte an dem gleichen Netzwerk vom Typ PROFIBUS, PROFINET, MPI oder ASI angeschlossen sind.

Die Verbindungen zwischen einem Netzwerk und den Geräteelementen werden als Referenz für das Netzwerkobjekt abgebildet. Es gibt keine Referenz von den Netzwerkobjekten zu den verbundenen Geräteelementen. Die Netzwerkparameter werden im Netzwerkobjekt gespeichert. Die Parameter, die eine Netzwerkschnittstelle eines bestimmten Geräteelements betreffen, das mit einem Netzwerk verbunden ist, werden in einem Netzwerkknotenobjekt in diesem Geräteelement gespeichert. Die Kommunikation wird oft mit „Kanälen“, „Ports“ und „Schnittstellen“ geregelt.

Subnetze werden als interne Elemente mit der Rollenklassifizierung „Subnetz“ in der Instanzhierarchie in der AML-Datei exportiert.

Ein Subnetz hat folgende verbundene Elemente in der AML-Struktur:

- Internes Element mit der Rollenklassifizierung „Knoten“
  Legt die Schnittstelle auf einem Geräteelement fest.
- `<InternalLink>`
  Legt die verbundenen Partner des Subnetzes fest. Der `<InternalLink>` Variablenname ist einzigartig und wird immer unter dem internen Element des Projekts in der AML-Datei hinzugefügt.

```
    ....
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/AutomationProject" />
    <InternalLink Name="Link To Port_1"
      RefPartnerSideA="1e3e4c5b-04c1-4d2c-9aee-cad53cc92dba:CommunicationPortInterface"
      RefPartnerSideB="d45aa36a-a7f2-4862-a266-d6727b9cfd75:CommunicationPortInterface" />
    <InternalLink Name="Link To Subnet_1"
      RefPartnerSideA="beb4eb8e-1a45-45ce-a703-1acfac73e5f3:LogicalEndPoint_Node"
      RefPartnerSideB="1062a384-d3ca-4183-9ac2-0934a5ab7286:LogicalEndPoint_Subnet" />
    <InternalLink Name="Link To Subnet_2"
      RefPartnerSideA="a3e85aed-580a-45c8-943e-da7de8280b7c:LogicalEndPoint_Node"
      RefPartnerSideB="1062a384-d3ca-4183-9ac2-0934a5ab7286:LogicalEndPoint_Subnet" />
  </InternalElement>
 </InstanceHierarchy>
</CAEXFile>
```

- `<ExternalInterface>`
  Stellt in internen Elementen von Knoten und Subnetzen fest, dass Knoten und Subnetze verbunden sind. Wenn die Knoten oder Subnetze nicht verbunden sind, existieren die `<ExternalInterface>` Elemente für Knoten und Subnetz nicht.

```
...
<InternalElement ID="1062a384-d3ca-4183-9ac2-0934a5ab7286" Name="PN/IE_1">
  <Attribute Name="Type" AttributeDataType="xs:string">
    <Value>Ethernet</Value>
  </Attribute>
  <ExternalInterface ID="55ecf2cb-e72a-4974-8ed0-1d4e1ade3509"
      Name="LogicalEndPoint_Subnet"
      RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
  <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Subnet" />
</InternalElement>
...
```

Verwendung

Das rechnergestützte Importieren/Exportieren unterstützt folgende Arten von Subnetzen:

- Ethernet
- PROFIBUS
- MPI
- ASi

Attribute eines „Subnetz“-Elements

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	
Type	Obligatorisch	Ethernet oder PROFIBUS oder MPI oder ASi

Attribute von Elementen "CommunicationInterface"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte für CAx-Import- und -Exportdateien:

Attribut	Verarbeitung	Kommentar
Name	Obligatorisch	Keine Relevanz bei "festen" Geräteelementen.
Label	Obligatorisch	Beschriftung fehlt möglicherweise, wenn "BuiltIn" = TRUE und "PositionNumber" für das zugehörige Objekt "DeviceItem" angegeben sind.
TypeIdentifier	Obligatorisch	
FirmwareVersion	Obligatorisch	
TypeName	Nur Export	Keine Relevanz bei "integrierten" Geräteelementen.
DeviceItemType	Nur Export	Nur für CPU und Kopfmodul
PositionNumber	Obligatorisch	Keine Relevanz beim Import von "integrierten" Geräteelementen.
BuiltIn	Obligatorisch für Export Optional für Import	Keine Relevanz beim Import von "nicht integrierten" Geräteelementen. Für den Import standardmäßig "False".
Comment	Optional	Nicht zutreffend bei "integrierten" Geräteelementen.

Attribute von Elementen "CommunicationPort"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	Keine Relevanz bei "integrierten" Geräteelementen.
Label	Obligatorisch	Das Label kann fehlen, wenn „BuiltIn" = TRUE“ und „PositionNumber“ für das zugehörige „DeviceItem“-Objekt festgelegt sind.

Attribut	Handhabung	Kommentar
TypeIdentifier	Obligatorisch	
FirmwareVersion	Obligatorisch	
TypeName	Nur Export	Keine Relevanz bei "integrierten" Geräteelementen.
DeviceItemType	Nur Export	Nur für CPU und Kopfmodul.
PositionNumber	Obligatorisch	Nur relevant beim Import von "integrierten" Geräteelementen, wenn das Attribut "Label" nicht angegeben ist. Wenn "PositionNumber" und "Label" beide konfiguriert sind, erhält "PositionNumber" die höhere Priorität.
BuiltIn	Obligatorisch für Export Optional für Import	Keine Relevanz beim Import von "nicht integrierten" Geräteelementen. Für den Import standardmäßig "False".
Comment	Optional	Nicht zutreffend bei "integrierten" Geräteelementen.

Attribute eines „Knoten“-Elements

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

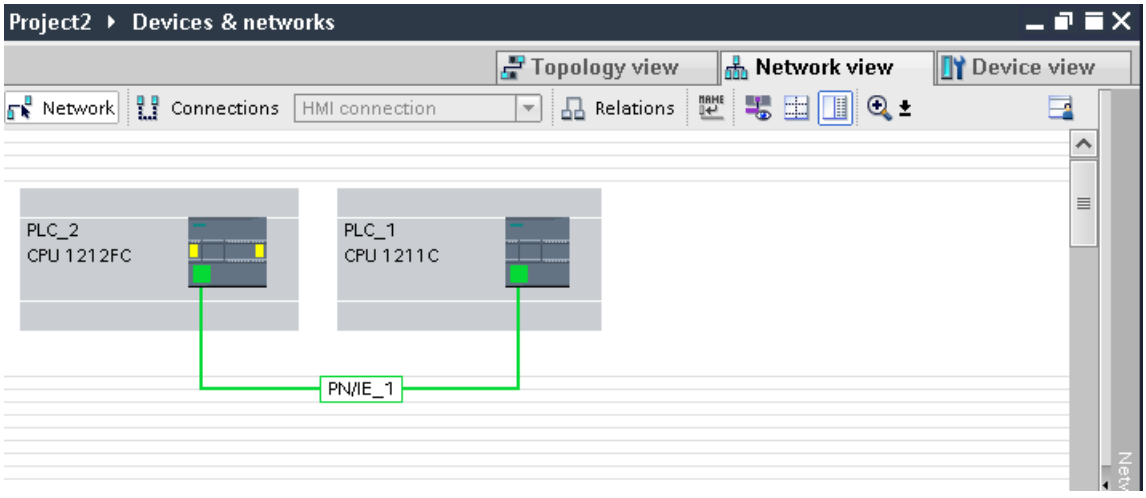
Attribut	Handhabung	Kommentar
Name	Nur Export	MPI, PROFIBUS, PROFINET
Type	Nur Exportieren	Ethernet oder PROFIBUS oder MPI oder ASi
NetworkAddress	Obligatorisch	
SubnetMask	Optional	PROFINET Beim Import wird der Standardwert beibehalten, wenn kein Wert festgelegt ist.
RouterAddress	Optional	PROFINET Beim Import wird der Standardwert beibehalten, wenn kein Wert festgelegt ist.
DhcpClientId	Optional	PROFINET Beim Import wird der Standardwert beibehalten, wenn kein Wert festgelegt ist.
IpProtocolSelection	Optional	PROFINET Beim Import wird der Standardwert beibehalten, wenn kein Wert festgelegt ist. Werte: Project, Dhcp, UserProgram, OtherPath

Attribute eines „Kanal“-Elements

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Type	Obligatorisch	Digital oder analog
IoType	Obligatorisch	Eingang oder Ausgang
Number	Obligatorisch	
Length	Nur Exportieren	

Beispiel: Exportiertes Subnetz



AML-Struktur

Die folgenden Abbildungen zeigen die Struktur der exportierten AML-Datei.

```
...
<InstanceHierarchy Name="APC Sample Instance Hierarchy">
  <InternalElement ID="e9d2bedb-f8c1-4148-acda-c3c68836c7dd" Name="Project2">
    ...
      <InternalElement ID="1062a384-d3ca-4183-9ac2-0934a5ab7286" Name="PN/IE_1">
      <Attribute Name="Type" AttributeDataType="xs:string">
        <Value>Ethernet</Value>
      </Attribute>
      <ExternalInterface ID="55ecf2cb-e72a-4974-8ed0-1d4e1ade3509"
          Name="LogicalEndPoint_Subnet"
              RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
      <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Subnet" />
    </InternalElement>
    <InternalElement ID="b011dbb1-efa4-46c0-a26f-f9bd047cda4f" Name="S7-1200 station_1">
          ...
          <InternalElement ID="d006e41b-05ff-44ab-baab-fca15f99e86c" Name="PROFINET interface_1">
            ...
            <InternalElement ID="beb4eb8e-1a45-45ce-a703-1acfac73e5f3" Name="E1">
              ...
              <ExternalInterface ID="a365b498-20cc-4e0b-99ca-5c5257632b96"
                  Name="LogicalEndPoint_Node" RefBaseClassPath=
                      "CommunicationInterfaceClassLib/LogicalEndPoint" />
              <SupportedRoleClass RefRoleClassPath=
                  "AutomationProjectConfigurationRoleClassLib/Node" />
            </InternalElement>
            <InternalElement ID="d45aa36a-a7f2-4862-a266-d6727b9cfd75" Name="Port_1">
              ...
              <ExternalInterface ID="32c6ba4a-b01f-4678-b721-ea284779e96c"
                  Name="CommunicationPortInterface"
                  RefBaseClassPath=
                  "AutomationProjectConfigurationInterfaceClassLib/CommunicationPortInterface" />
              <SupportedRoleClass RefRoleClassPath=
              "AutomationProjectConfigurationRoleClassLib/CommunicationPort" />
            </InternalElement>
            <SupportedRoleClass RefRoleClassPath=
              "AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
          </InternalElement>
          <SupportedRoleClass RefRoleClassPath=
              "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
        </InternalElement>
        <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
      </InternalElement>
      <SupportedRoleClass RefRoleClassPath=
          "AutomationProjectConfigurationRoleClassLib/Device" />
    </InternalElement>
  ...
```

```
...
<InternalElement 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</...
    <InternalElement ID="a3e85aed-580a-45c8-943e-da7de8280b7c" Name="E1">
      <Attribute Name="Type" AttributeDataType="xs:string">
        <Value>Ethernet</Value>
      </Attribute>
      <Attribute Name="NetworkAddress" AttributeDataType="xs:string">
        <Value>192.168.0.2</Value>
      </Attribute>
      <Attribute Name="SubnetMask" AttributeDataType="xs:string">
        <Value>255.255.255.0</Value>
      </Attribute>
      <Attribute Name="DeviceNumber" AttributeDataType="xs:string">
        <Value>0</Value>
      </Attribute>
      <Attribute Name="IpProtocolSelection" AttributeDataType="xs:string">
        <Value>Project</Value>
      </Attribute>
      <ExternalInterface ID="6ae8eb93-09d3-4f8c-b529-a12148c71bf4"
          Name="LogicalEndPoint_Node"
          RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
      <SupportedRoleClass RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
    </InternalElement>
    <InternalElement ID="1e3e4c5b-04c1-4d2c-9aee-cad53cc92dba" Name="Port_1">
      ...
      <ExternalInterface ID="1f5b2a3d-fcd1-460a-b846-30dadc8726d1"
          Name="CommunicationPortInterface"
          RefBaseClassPath=
          "AutomationProjectConfigurationInterfaceClassLib/CommunicationPortInterface" />
      <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/CommunicationPort" />
    </InternalElement>
    <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
      "AutomationProjectConfigurationRoleClassLib/DeviceItem" />
</InternalElement>
...
```

Siehe auch

Struktur der CAx-Daten zum Import/Export (Seite 566)

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

8.5.20 Exportieren/Importieren von PLC-Variablen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts
- Die PLC ist offline.

Verwendung

Exportierte und importierte Symbole und Variablen werden einem Geräteelement zugeordnet. Rechnergestütztes Importieren/Exportieren betrifft hardwarenahe Symbole und Variablen. Die Symbole und Variablen werden nur mit dem Geräteelement des Steuersollwerts exportiert, z. B. der CPU und nicht mit anderen Geräteelementen, auf die sie sich beziehen, z. B. einem E/A-Modul. Wie Geräte, werden die Variablen oft in Variablentabellen und in einer hierarchischen Ordnerstruktur unterteilt.

AML-Strukturelemente

PLC-Variablen, Variablentabellen und Variablen-Nutzerordner können durch die rechnergestützte Import-/Exportfunktion exportiert und importiert werden. Die Variablen-Objekte werden in den folgenden AML-Strukturelementen abgebildet:

- `<InternalElement>`
  Variablentabellen und Variablen-Nutzerordner werden als interne Elemente der zugehörigen PLC mit der jeweiligen Rollenklassifizierung abgebildet.
- `<ExternalInterface>`
  Stellt eine PLC-Variable dar, die zu dem internen Element der zugehörigen Variablentabelle oder dem Variablen-Nutzerordner gehört.

Ein Zuordnungskanal mit einer PLC-Variable wird über das Element `<internal link>` als Kommunikationspartner exportiert. Die folgende XML-Struktur zeigt ein Beispiel:

```
...
    <InternalLink Name="Link To Tag_1"
       RefPartnerSideA="b33451f6-d88f-4900-8dbe-41f1be1e3535:Channel_DI_0"
       RefPartnerSideB="b2b937ee-d5db-4826-9340-027b1da22828:Tag_1" />
...
```

PLC-Variablen-Nutzerordner

Die Objekte „TagUserFolder“ benötigen nur das Attribut „Name“ in rechnergestützten Import- und Exportdateien.

Attribute einer PLC-Variablentabelle

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch, wird ignoriert, wenn „AssignToDefault" “ = TRUE:	
AssignToDefault	Nur Importieren	Wird verwendet, um die Standardvariablentabelle während des Imports zu identifizieren. Wenn "AssignToDefault" = TRUE, werden alle Variablen im Rahmen der Standardvariablentabelle des TIA Portals erstellt.

Attribute einer PLC-Variablen

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	
DataType	Obligatorisch	
LogicalAddress	Obligatorisch	Wird im internationalen Mnemonik-Format importiert und exportiert
Comment	Optional	

Beispiel: AML-Struktur

Die folgende Abbildung zeigt die Struktur der folgenden exportierten Variablenobjekte:

- leere Standardvariablentabelle
- Variablen-Nutzerordner „Gruppe_1“
- enthaltene Variablentabelle „Variablentabelle_1“
- vier Variablen

```
...
<InternalElement ID="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>IW0</Value>
      </Attribute>
    </ExternalInterface>
    <SupportedRoleClass RefRoleClassPath=
       "AutomationProjectConfigurationRoleClassLib/TagTable" />
  </InternalElement>
  <SupportedRoleClass RefRoleClassPath=
    "AutomationProjectConfigurationRoleClassLib/TagUserFolder" />
</InternalElement>
...
```

Siehe auch

Struktur der CAx-Daten zum Import/Export (Seite 566)

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

8.5.21 Export/Import von IO-Systemen

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts
- Die PLC ist offline.

AML-Struktur

IO-Systeme werden in der AML-Struktur als `<InternalElement>` dargestellt.

IO-Systeme eines Masters oder IO-Controllers werden unter dem Element `<CommunicationInterface>` des Geräteelements einer Schnittstelle hinzugefügt.

```
...
<InternalElement ID="[Communication Interface UniqueID]"
      Name="[Communication Interface Name]">
...
  <!--Node-->
  <InternalElement ID="[Node UniqueID]" Name="[Node Name]">
    ...
    <ExternalInterface ID="[External Interface UniqueID]"
      Name="LogicalEndPoint_Node"
      RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
    <SupportedRoleClass
      RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
  </InternalElement>
  <!--IoSystem-->
  <InternalElement ID="[IoSystem UniqueID]" Name="[IoSystem Name]">
   <Attribute Name="Number" AttributeDataType="xs:integer">
    <Value>[IoSystem Number]</Value>
    </Attribute>
    <ExternalInterface ID="[External Interface UniqueID]"
      Name="LogicalEndPoint_Interface"
      RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
    <SupportedRoleClass
      RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/IoSystem" />
  </InternalElement>
  <SupportedRoleClass
      RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
</InternalElement>
```

Als Slave oder IO-Device verbundene IO-Systeme werden als Elemente `<ExternalInterface>` unter dem Element `<CommunicationInterface>` des Geräteelements einer Schnittstelle hinzugefügt.

```
<InternalElement ID="[Communication Interface UniqueID]"
    Name="[Communication Interface Name]">
 ...
  <ExternalInterface ID="[External Interface UniqueID]"
    Name="LogicalEndPoint_Interface"
    RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
  <!--Node-->
  <InternalElement ID="[Node UniqueID]" Name="[Node Name]">
    <ExternalInterface ID="[External Interface UniqueID]"
    Name="LogicalEndPoint_Node"
    RefBaseClassPath="CommunicationInterfaceClassLib/LogicalEndPoint" />
    <SupportedRoleClass
    RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/Node" />
  </InternalElement>
  <SupportedRoleClass
    RefRoleClassPath="AutomationProjectConfigurationRoleClassLib/CommunicationInterface" />
</InternalElement>
```

Die verbundenen Partner des IO-Systems werden als Elemente `<InternalLink>` dargestellt. Die Variablen `<InternalLink>` werden unter dem gemeinsamen

übergeordneten Element eines IO-Systems und des verbundenen Slave-Geräteelements hinzugefügt, z. B. unter Project, DeviceFolder, DeviceItem.

Der Variablenname `<InternalLink>` ist im gemeinsamen übergeordneten Element eindeutig.

Attribute eines Elements "IO-System"

Die folgende Tabelle zeigt die zugehörigen Attribute der Objekte von rechnergestützten Import- und Exportdateien:

Attribut	Handhabung	Kommentar
Name	Obligatorisch	Der Name des IO-Systems. Wird ein leerer String importiert, wird das IO-System mit dem Standardnamen erstellt.
Number	Optional	Erfolgt keine Angabe für den Import, wird der Standardwert verwendet.

8.5.22 Exportieren/Importieren mehrsprachiger Kommentare

Voraussetzung

- Die TIA Portal Openness-Anwendung ist mit dem TIA Portal verbunden.
  Siehe Verbindung zum TIA Portal herstellen
- Ein Projekt ist geöffnet.
  Siehe Öffnen eines Projekts
- Die PLC ist offline.

Verwendung

Die rechnergestützten Export- und Importkommentare und mehrsprachigen Kommentare zum Datenaustausch der folgenden Hardwareobjekte:

- Geräte (Gerät)
- Module (Geräteelemente)
- Variablen (Variable)

Das Importieren/Exportieren mehrsprachiger Kommentare umfasst alle Sprachen des TIA Portals.

Einschränkungen

- Export
  - Nur, wenn ein Kommentar existiert, wird ein Attribut „Comment“ in die AML-Datei exportiert.
- Import
  - Das Attribut "Comment" ist optional.
  - Für virtuelle Geräteelemente kann kein Kommentar importiert werden.

Beispiel: Exportierte Konfigurierung mit mehrsprachigen Kommentaren

Das folgende Bild zeigt die Konfigurierung eines SIMATIC S7 1500 (Gerät) mit PLC_1 (Geräteelemente). Bei beiden Objekten werden Kommentare in Englisch, Französisch, Deutsch und Chinesisch festgelegt.

AML-Struktur

Nach dem Exportieren dieser Konfigurierungen, werden die mehrsprachigen Kommentare als verschachtelte Attribute des Geräts, Geräteelements oder der Variablen generiert.

- Das übergeordnete Attribut "Comment" muss den in der Standardsprache verwendeten Wert besitzen.
- Das Nachfolgerattribut existiert für jeden fremdsprachigen Kommentar.

```
...
<Attribute Name="Comment" AttributeDataType="xs:string">
  <Value>English</Value>
  <Attribute Name="aml-lang=en-US" AttributeDataType="xs:string">
    <Value>English</Value>
  </Attribute>
  <Attribute Name="aml-lang=de-DE" AttributeDataType="xs:string">
    <Value>Deutsch</Value>
  </Attribute>
  <Attribute Name="aml-lang=zh-HK" AttributeDataType="xs:string">
    <Value>Chinese</Value>
  </Attribute>
  ...
  ...
</Attribute>
...
```

Siehe auch

Struktur der CAx-Daten zum Import/Export (Seite 566)

Verbindung zum TIA Portal aufbauen (Seite 74)

Projekt öffnen (Seite 99)

8.5.23 AML-Attribute im Vergleich zu TIA Portal Openness-Attributen

Auf Attribute und Export-/Importattribute zugreifen

Über TIA Portal Openness können Sie auf Hardware-Attribute von Hardwareobjekten zugreifen. Einzelne Namen, die Sie für den Zugriff auf diese Attribute verwenden, z. B. von einem Geräteelement, unterscheiden sich von den Attributnamen in der Export-/Import-AML-Datei.

Liste der Attribute

Die folgende Tabelle bietet einen Überblick über beide Arten von Attributen:

Tabelle 8-6 Attributnamen von Geräten und GSD/GSDML-Geräten

AML-Datei	TIA Portal Openness
Name	Name
TypeIdentifier	TypeIdentifier
Comment	Comment

Tabelle 8-7 Attributnamen von Geräteelementen

AML-Datei	TIA Portal Openness
Name	Name
TypeIdentifier	Zugeordnet zum Substring von <TypeIdentifier> (d.h. Wert vor ersten Operator "/"), wobei der Bestandteil mit der Firmwareversion ignoriert wird. Die Zuordnung des Substrings ist nur zutreffend, wenn die Typkennung mit dem Präfix <OrderNumber:> beginnt und einen Bestandteil mit der Firmwareversion hat, ansonsten erfolgt die Zuordnung zur Vervollständigung von <TypeIdentifier>.
FirmwareVersion	<FirmwareVersion> ist dem Substring von <TypeIdentifier> zugeordnet (d. h. Wert nach dem ersten Operator "/"). Die Zuordnung des Substrings ist nur zutreffend, wenn <TypeIdentifier> mit dem Präfix <OrderNumber:> beginnt und einen Bestandteil mit der Firmwareversion hat.
TypeName	TypeName
DeviceItemType (für CPU und Kopfmodul)	**Classification**
PositionNumber	PositionNumber
BuiltIn	**IsBuiltIn**
PlantDesignation IEC	**PlantDesignation**
LocationIndentifier IEC	**LocationIdentifier**
Comment	Comment

Tabelle 8-8 Attributnamen von GSD/GSDML-Geräteelementen

AML-Datei	TIA Portal Openness
Name	Name
TypeIdentifier	TypeIdentifier
TypeName	TypeName
DeviceItemType (für Kopfmodul)	**Classification**
PositionNumber	PositionNumber
BuiltIn	**IsBuiltIn**
Comment	Comment
Label	Label

Tabelle 8-9 Attributnamen von Variablen

AML-Datei	TIA Portal Openness
Name	Name
DataType	**DataTypeName**

AML-Datei	TIA Portal Openness
LogicalAddress	LogicalAddress
Comment	Comment

Tabelle 8-10 Attributnamen von Variablentabellen

AML-Datei	TIA Portal Openness
Name	Name
AssignToDefault	**IsDefault**

Tabelle 8-11 Attributnamen von Adressen

AML-Datei	TIA Portal Openness
StartAddress	StartAddress
Length	Length
IoType	IoType

Tabelle 8-12 Attributnamen von Ports

AML-Datei	TIA Portal Openness
Name	Name
TypeIdentifier	TypeIdentifier
FirmwareVersion	FirmwareVersion
TypeName	TypeName
PositionNumber	PositionNumber
BuiltIn	**IsBuiltIn**
Comment	Comment
Label	Label

Tabelle 8-13 Attributnamen von Geräten mit IO-Schnittstelle

AML-Datei	TIA Portal Openness
Name	Name
TypeIdentifier	TypeIdentifier
FirmwareVersion	FirmwareVersion
TypeName	TypeName
DeviceItemType (für CPU und Kopfmodul)	**Classification**
PositionNumber	PositionNumber
BuiltIn	**IsBuiltIn**
Label	Label
Comment	Comment

Tabelle 8-14 Attributnamen von Kanälen

AML-Datei	TIA Portal Openness
Type	Type
IoType	IoType
Number	Keinem Attribut in TIA Portal Openness zugeordnet.
Length	ChannelWidth

Wesentliche Änderungen

9

9.1 Größere Änderungen in TIA Portal Openness V15

Änderungen

Wenn Sie die Hinweise zur versionsübergreifenden Programmierung beachtet haben und Ihr Projekt nicht auf V15 aktualisieren, laufen Ihre Anwendungen ohne jede Einschränkung auf jedem Rechner, selbst wenn nur ein TIA Portal V15 installiert ist.

Wenn Sie Ihr Projekt auf V15 aktualisieren, ist es notwendig, Ihre Anwendung mit der SiemensEngineering.dll von V15 neu zu übersetzen.

In manchen Fällen ist es notwendig, den Code Ihrer Anwendung anzupassen

- Verhaltensänderungen für Zusammensetzungen in DeviceItemComposition
- BitOffset von ASi-Adressen
- Klasse Exception
- Systemordner von System-UDTs
- Submodule haben nicht die Attribute Author und TypeName
- Zeitstempel für letzte Änderung
- Export-XML für GRAPH-Bausteine
- Variablentabellen importieren
- Nicht fehlersichere relevante Attribute eines PLC ändern
- F-Parameter bei eingestelltem Sicherheitspasswort ändern
- Zugriff auf Objekte in einer S7-1200 CPU

Verhaltensänderungen für Zusammensetzungen in DeviceItemComposition

Die folgenden Zusammensetzungen in DeviceItemCompositon wurden geändert, um ein dynamisches Verhalten zu erreichen: Die Zusammensetzung wird jetzt aktualisiert, wenn ein Element über die Benutzeroberfläche des TIA Portals hinzugefügt oder gelöscht wird.

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

BitOffset von ASi-Adressen

Wenn ein Modul eine Eingangs- und eine Ausgangsadresse für beide Adressobjekte hat, wird das korrekte Attribut BitOffset bereitgestellt.

Wenn ein Modul Kanäle hat, wird das Attribut BitOffset für den Kanal nicht bereitgestellt.

Klasse Exception

ServiceID und MessageID wurden aus der Klasse exception entfernt.

Submodule haben nicht die Attribute Author und TypeName.

Die Attribute Author und TypeName wurden von Submodulen, die nicht gesteckt werden können, entfernt.

Systemordner von System-UDTs

Bei Systemordnern von System-UDTs werden der entsprechende Ordner und die entsprechende Zusammensetzung bereitgestellt. Das führt auch zu einer Änderung in der Hierarchie von Vergleichsergebnissen.

Zeitstempel für letzte Änderung

Wenn während eines Upgrade ein Objekt geändert wird, ändert sich auch der Zeitstempel für die letzte Änderung.

Export-XML für GRAPH-Bausteine

Die Export-XML für GRAPH-Bausteine enthält eine zusätzliche leere Aktion: <Actions />.

Variablentabellen importieren

Die Einstellung von Variablenattributen ist nicht mehr von Datentypen abhängig.

Nicht fehlersichere relevante Attribute eines PLC ändern

Alle nicht fehlersicheren relevanten Attribute eines PLC können über TIA Portal Openness geändert werden, selbst wenn ein Sicherheitspasswort eingestellt ist.

F-Parameter bei eingestelltem Sicherheitspasswort ändern

F-Parameter von F-IO können nur geändert werden, wenn das Sicherheitspasswort nicht eingestellt ist.

Zugriff auf Objekte in einer S7-1200 CPU

Der Zugriff auf Array-Variablen für die Technologieobjekte TO_PositioningAxis und TO_CommandTable wurde geändert. Einzelheiten hierzu finden Sie im Kapitel über S7-1200 Motion Control.

9.2 Die wichtigsten Änderungen in V14 SP1

9.2.1 Die wichtigsten Änderungen in V14 SP1

Einleitung

Die folgenden Änderungen wurden bei dem TIA Portal Openness API-Objektmodell V14 SP1 vorgenommen, was möglicherweise Auswirkungen auf Ihre vorhandenen Anwendungen hat:

Änderung	Erforderliche Anpassung von Programmcode
Verbesserte Handhabung von Masterkopien	Die Aktion CreateFrom erstellt ein neues Objekt basierend auf einer Masterkopie in einer Bibliothek und platziert es in der Zusammensetzung, wo die Aktion aufgerufen wurde. Die Aktion CreateFrom unterstützt nur Masterkopien, die ausschließlich einzelne Objekte enthalten. Der Ausgabetyp entspricht dem jeweils zusammengesetzten Typ. Die folgenden Zusammensetzungen unterstützen CreateFrom: • Siemens.Engineering.HW.DeviceComposition • Siemens.Engineering.HW.DeviceItemComposition • Siemens.Engineering.SW.Blocks.PlcBlockComposition • Siemens.Engineering.SW.Tags.PlcTagTableComposition • Siemens.Engineering.SW.Tags.PlcTagComposition • Siemens.Engineering.SW.Types.PlcTypeComposition • Siemens.Engineering.SW.TechnologicalObjects.TechnologicalInstanceDBComposition • Siemens.Engineering.SW.Tags.PlcUserConstantComposition • Siemens.Engineering.Hmi.Tag.TagTableComposition • Siemens.Engineering.Hmi.Tag.TagComposition • Siemens.Engineering.Hmi.Screen.ScreenComposition • Siemens.Engineering.Hmi.Screen.ScreenTemplateComposition • Siemens.Engineering.Hmi.RuntimeScripting.VBScriptComposition • Siemens.Engineering.HW.SubnetComposition • Siemens.Engineering.HW.DeviceUserGroupComposition • Siemens.Engineering.SW.Blocks.PlcBlockUserGroupComposition • Siemens.Engineering.SW.ExternalSources.PlcExternalSourceUserGroupComposition • Siemens.Engineering.SW.Tags.PlcTagTableUserGroupComposition • Siemens.Engineering.SW.Types.PlcTypeUserGroupComposition
Verbesserte Handhabung globaler Bibliotheken	Bestehende Aktionen mit globalen Bibliotheken können jetzt Änderungsaktionen umfassen, z. B. Löschen einer Masterkopie aus einer globalen Bibliothek. UpdateProject und UpdateLibrary verwenden nicht länger die Parameter UpdatePathsMode und DeleteUnusedVersionsMode. Nicht verwendete Versionen werden nach einem Update nicht gelöscht

Änderung	Erforderliche Anpassung von Programmcode
Änderung von System.String in System.IO.FileInfo Änderung von System.String in System.IO.DirectoryInfo	Alle Ereignisse, bei denen ein String-Pfad spezifiziert werden muss, verwenden den Pfad FileInfo oder den Pfad DirectoryInfo. Beispiel: • Projekt öffnen • Bibliothek öffnen • Projekt erstellen • Globale Bibliothek erstellen • ...

Neue Elemente im Objektmodell

Name	Typ	Namensraum	Kommentar
PlcUserConstant	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstant
PlcUserConstantComposition	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstantComposition
PlcSystemConstant	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstant
PlcSystemConstantComposition	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstantComposition
MultilingualTextItem	Klasse	Siemens.Engineering	Auf mehrsprachige Texte zugreifen.
MultilingualTextItemComposition	Klasse	Siemens.Engineering	Auf mehrsprachige Texte zugreifen.
TiaPortalTrustAuthority.FeatureTokens	Enumerationswert	Siemens.Engineering	Auf Einstellungen des TIA Portals zugreifen.
TiaPortalSetting	Klasse	Siemens.Engineering.Settings	Auf Einstellungen des TIA Portals zugreifen.
TiaPortalSettingComposition	Klasse	Siemens.Engineering.Settings	Auf Einstellungen des TIA Portals zugreifen.
TiaPortalSettingsFolder	Klasse	Siemens.Engineering.Settings	Auf Einstellungen des TIA Portals zugreifen.
TiaPortalSettingsFolderComposition	Klasse	Siemens.Engineering.Settings	Auf Einstellungen des TIA Portals zugreifen.
LanguageAssociation	Klasse	Siemens.Engineering	Auf aktive Sprachen zugreifen.
LanguageComposition.Find	Methode	Siemens.Engineering	Auf aktive Sprachen zugreifen.

Geänderte Elemente im Objektmodell

Name	Typ	Namensraum	Kommentar
PlcConstant	Klasse	Siemens.Engineering.SW.Tags	Herausgegebene Basisklasse von PlcUserConstant und PlcSystemConstant.
PlcTag	Klasse	Siemens.Engineering.SW.Tags	Teilen von PlcConstantComposition
ITargetComparable	Schnittstelle	Siemens.Engineering.Compare	String-Attribut DataTypeName anstelle eines offenen Links DataType.
MultilingualText	Klasse	Siemens.Engineering	Auf mehrsprachige Texte zugreifen.

Name	Typ	Namensraum	Kommentar
ProjectComposition.Create	Methode	Siemens.Engineering	Parameter geändert zur Verwendung von DirectoryInfo und eines Strings.
Project.Subnets	Attribut	Siemens.Engineering	Auf Subnetze zugreifen
Project.Languages	Attribut	Siemens.Engineering	Verschoben, um Attribut von Siemens.Engineering.LanguageSettings darzustellen und unterstützte Sprachen bereitzustellen

Gelöschte Elemente im Objektmodell

Name	Typ	Namensraum	Kommentar
PlcConstantComposition	Klasse	Siemens.Engineering.SW.Tags	Teilen in PlcSystemConstantComposition und PlcUserConstantComposition.
CompareResultElement.PathInformation	Attribut	Siemens.Engineering.SW.Tags	Nicht mehr verwendet.
MultilingualText.GetText(CultureInfo cultureInfo)	Methode	Siemens.Engineering.Compare	Geändertes Konzept für den Zugriff auf Textelemente von MultilingualText.
TiaPortalTrustAuthority.CustomerIdentification	Enumerationswert	Siemens.Engineering	Nicht mehr verwendet.
TiaPortalTrustAuthority.ElevatedAccessExtensions	Enumerationswert	Siemens.Engineering	Nicht mehr verwendet.

Verhaltensänderungen

Name	Typ	Namensraum	Kommentar
PlcTag.Export(FileInfo path, ExportOptions options)	Methode	Siemens.Engineering.SW.Tags	Der Wert des Attributs LogicalAddress wird jetzt immer in internationaler Mnemonik exportiert. Deutsche Mnemonik wird für den Import noch akzeptiert.
PlcTag.LogicalAddress	Attribut	Siemens.Engineering.SW.Tags	Der Wert des Attributs LogicalAddress wird jetzt immer in internationaler Mnemonik zurückgegeben. Deutsche Mnemonik wird für Schreiben noch akzeptiert.

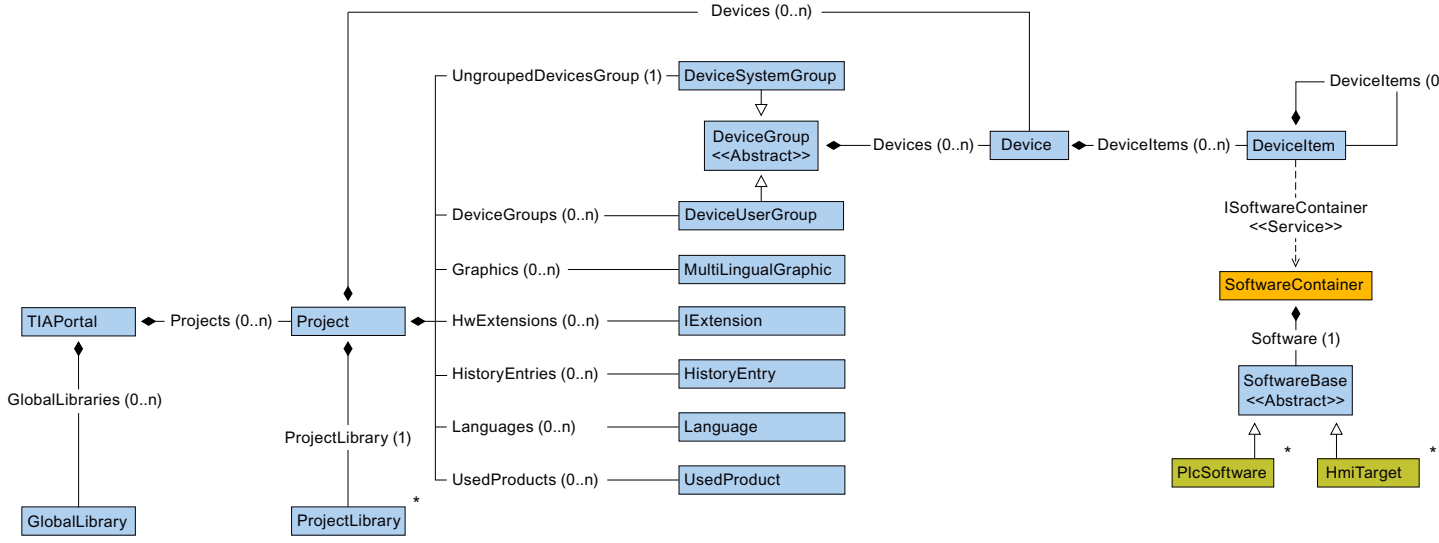
9.2.2 Wesentliche Änderungen im Objektmodell

Objektmodell von TIA Portal Openness V14

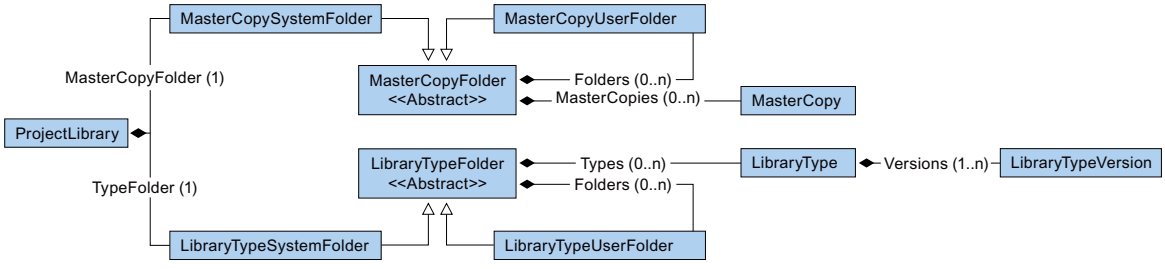
Um Ihnen einen Vergleich zwischen dem alten und dem neuen Objektmodell von TIA Portal Openness zu ermöglichen, beschreibt das nachstehende Diagramm das Objektmodell von TIA Portal V14.

Hinweis

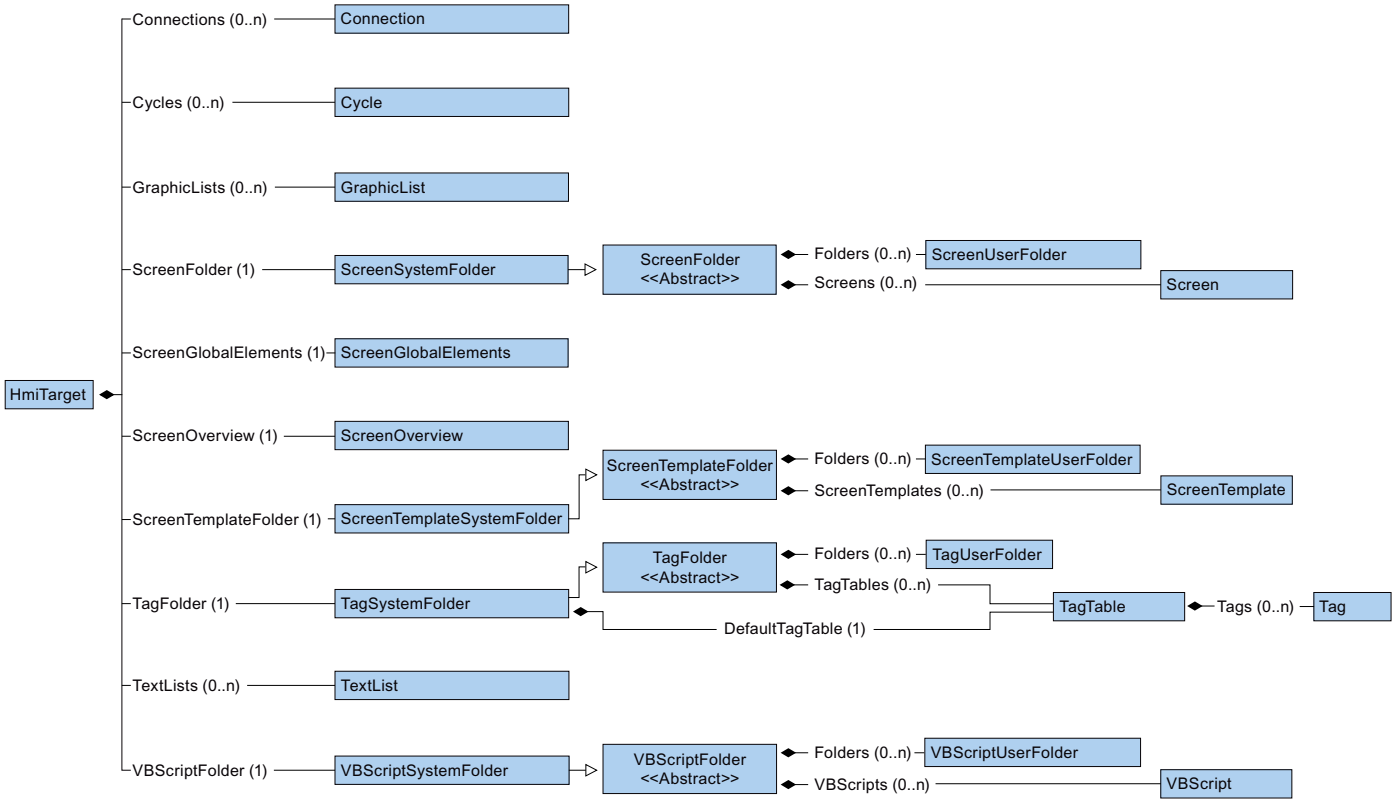
Das in dem Diagramm beschriebene Objektmodell ist veraltet. Sie finden Informationen über das Objektmodell von TIA Portal Openness V14 SP1 unter TIA Portal Openness-Objektmodell (Seite 51)



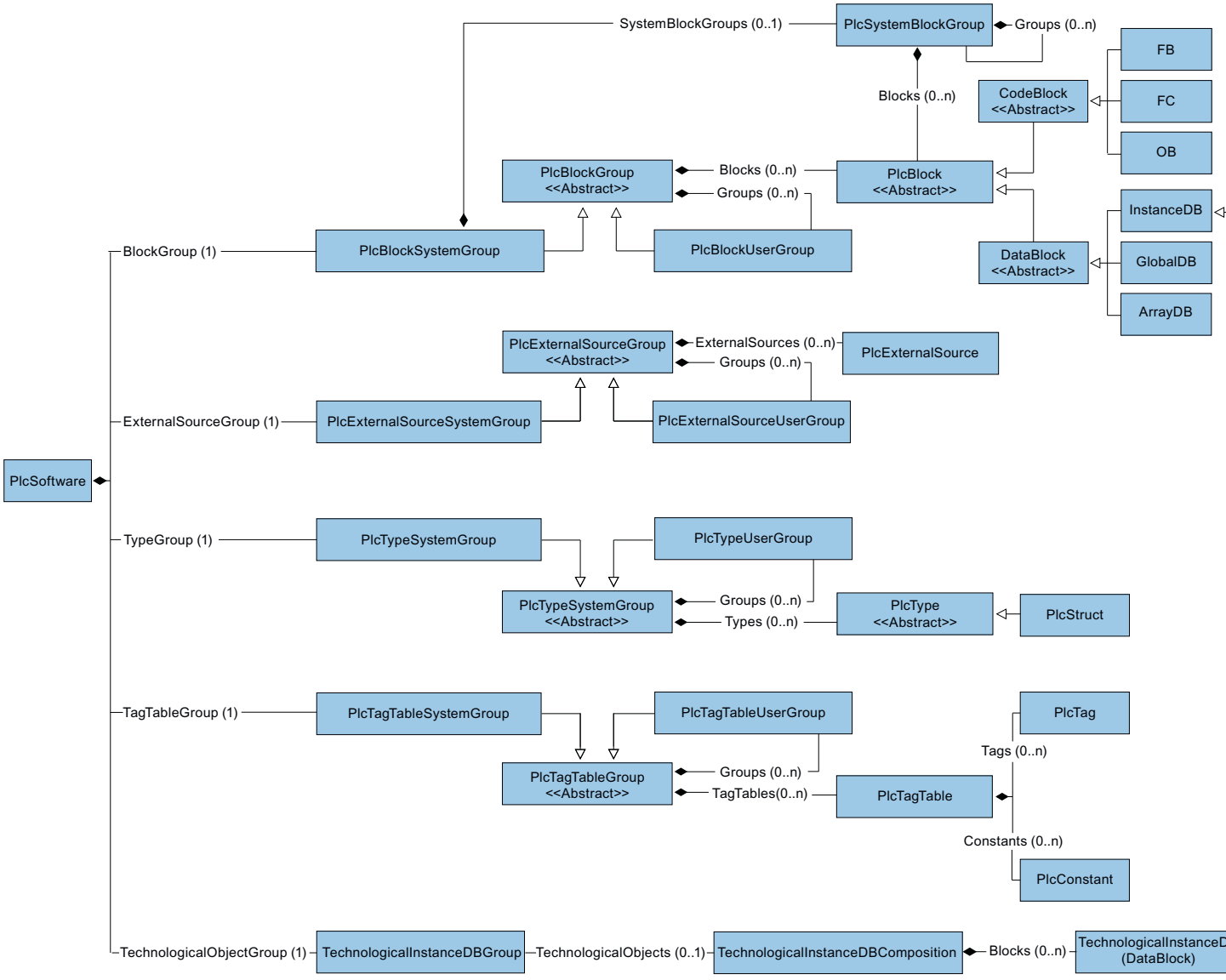
Das folgende Diagramm beschreibt die Objekte, die sich unter ProjectLibrary befinden.



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

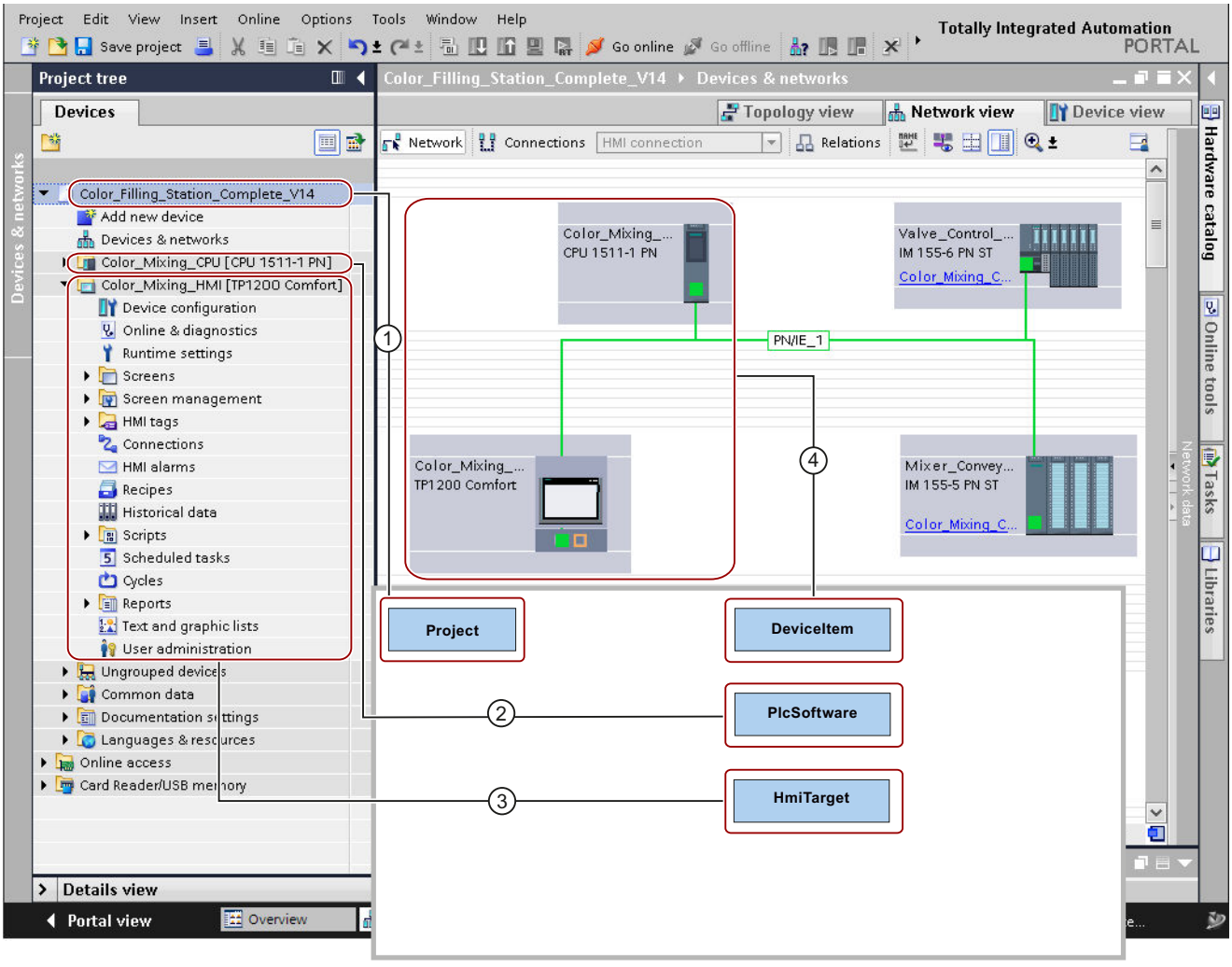


Das folgende Diagramm beschreibt die Objekte, die sich unter PlcSoftware befinden.


SystemBlockGroups (0..1)
PlcSystemBlockGroup
Groups (0..n)
FB
FC
OB
Blocks (0..n)
CodeBlock
<<Abstract>>
PlcBlockGroup
<<Abstract>>
Blocks (0..n)
Groups (0..n)
PlcBlock
<<Abstract>>
InstanceDB
GlobalDB
ArrayDB
DataBlock
<<Abstract>>
BlockGroup (1)
PlcBlockSystemGroup
PlcBlockUserGroup
PlcExternalSourceGroup
<<Abstract>>
ExternalSources (0..n)
Groups (0..n)
PlcExternalSource
ExternalSourceGroup (1)
PlcExternalSourceSystemGroup
PlcExternalSourceUserGroup
PlcSoftware
TypeGroup (1)
PlcTypeSystemGroup
PlcTypeUserGroup
PlcTypeSystemGroup
<<Abstract>>
Groups (0..n)
Types (0..n)
PlcType
<<Abstract>>
PlcStruct
TagTableGroup (1)
PlcTagTableSystemGroup
PlcTagTableUserGroup
PlcTag
Tags (0..n)
PlcTagTableGroup
<<Abstract>>
Groups (0..n)
TagTables(0..n)
PlcTagTable
Constants (0..n)
PlcConstant
TechnologicalObjectGroup (1)
TechnologicalInstanceDBGroup
TechnologicalObjects (0..1)
TechnologicalInstanceDBComposition
Blocks (0..n)
TechnologicalInstanceD
(DataBlock)

Beziehung zwischen TIA Portal und TIA Portal Openness-Objektmodell

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



① Das Objekt "Project" entspricht einem offenen Projekt im TIA Portal.

② Das Objekt "PlcSoftware" ist vom Typ "SoftwareBase"④ und entspricht einem PLC. Der Inhalt des Objekts entspricht einem PLC in der Projektnavigation mit Zugriff auf Objekte wie Bausteine oder PLC-Variablen.

③ Das Objekt "HmiTarget" ist vom Typ "SoftwareBase"④ und entspricht einem Bediengerät. Der Inhalt des Objekts entspricht einem HMI-Gerät in der Projektnavigation mit Zugriff auf Objekte wie Bilder oder HMI-Variablen.

④ Das Objekt "DeviceItem" entspricht einem Objekt im Editor "Geräte & Netze". Ein Objekt vom Typ "DeviceItem" kann ein Baugruppenträger oder ein Einschubmodul sein.

9.2.3 Änderungen an der Pilotfunktionalität

Einleitung

Die folgenden Änderungen wurden bei dem API-Objektmodell V14 SP1 vorgenommen und sind nur relevant für Anwender, die die Pilotfunktionalität von HW Konfig in V14 genutzt haben:

Änderungen an TIA Portal Openness API-Typen

TIA Portal Openness API-Typ	Neuer TIA Portal Openness API-Typ
Siemens.Engineering.HW.IAddress	Siemens.Engineering.HW.Address
Siemens.Engineering.HW.IAddressController	Siemens.Engineering.HW.Features.AddressController
Siemens.Engineering.HW.IChannel	Siemens.Engineering.HW.Channel
Siemens.Engineering.HW.IDevice	Siemens.Engineering.HW.Device
Siemens.Engineering.HW.IDeviceItem	Siemens.Engineering.HW.DeviceItem
Siemens.Engineering.HW.IExtension	Siemens.Engineering.HW.Extensions
Siemens.Engineering.HW.IGsd	Siemens.Engineering.HW.Features.GsdObject
Siemens.Engineering.HW.IGsdDevice	Siemens.Engineering.HW.Features.GsdDevice
Siemens.Engineering.HW.IGsdDeviceItem	Siemens.Engineering.HW.Features.GsdDeviceItem
Siemens.Engineering.HW.IHardwareObject	Siemens.Engineering.HW.HardwareObject
Siemens.Engineering.HW.IHwIdentifier	Siemens.Engineering.HW.HwIdentifier
Siemens.Engineering.HW.IHwIdentifierController	Siemens.Engineering.HW.Features.HwIdentifierController
Siemens.Engineering.HW.IIoConnector	Siemens.Engineering.HW.IoConnector
Siemens.Engineering.HW.IIoController	Siemens.Engineering.HW.IoController
Siemens.Engineering.HW.IIoSystem	Siemens.Engineering.HW.IoSystem
Siemens.Engineering.HW.IInterface	Siemens.Engineering.HW.Features.NetworkInterface
Siemens.Engineering.HW.Extensions.ModuleInformation-Provider	Siemens.Engineering.HW.Utilities.ModuleInformationProvider
Siemens.Engineering.HW.INode	Siemens.Engineering.HW.Node
Siemens.Engineering.HW.OPCUAExportProvider	Siemens.Engineering.HW.Utilities.OpcUaExportProvider
Siemens.Engineering.HW.IPort	Siemens.Engineering.HW.Features.NetworkPort
Siemens.Engineering.HW.IRole	Siemens.Engineering.HW.Features.HardwareFeature
	Siemens.Engineering.HW.Features.DeviceFeature
	Siemens.Engineering.HW.Utilities.ModuleInformationProvider
Siemens.Engineering.HW.SoftwareBase	Siemens.Engineering.HW.Software
Siemens.Engineering.HW.ISubnet	Siemens.Engineering.HW.Subnet
Siemens.Engineering.HW.ISoftwareContainer	Siemens.Engineering.HW.Features.SoftwareContainer
Siemens.Engineering.HW.ISubnetOwner	Siemens.Engineering.HW.Features.SubnetOwner

Änderungen für Enums

TIA Portal Openness API-Typ	Datentyp	Neuer TIA Portal Openness API-Typ	Datentyp
Siemens.Engineering.HW.Enums.AddressContext		Siemens.Engineering.HW.AddressContext	
Siemens.Engineering.HW.Enums.AddressIoType		Siemens.Engineering.HW.AddressIoType	
Siemens.Engineering.HW.Enums.Attachment-Type		Siemens.Engineering.HW.MediumAttachment-Type	
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.ChannelIoType	Byte	Siemens.Engineering.HW.ChannelIoType	Int
Siemens.Engineering.HW.Enums.ChannelType	Byte	Siemens.Engineering.HW.ChannelType	Int
Siemens.Engineering.HW.Enums.DeviceItemC-lassifications		Siemens.Engineering.HW.DeviceItemClassificati-ons	
Siemens.Engineering.HW.Enums.InterfaceOpera-tingModes		Siemens.Engineering.HW.InterfaceOperatingMo-des	
Siemens.Engineering.HW.Enums.IpProtocolSe-lection		Siemens.Engineering.HW.IpProtocolSelection	
Siemens.Engineering.HW.Enums.MediaRedun-dancyRole		Siemens.Engineering.HW.MediaRedundancyRole	
Siemens.Engineering.HW.Enums.NetType		Siemens.Engineering.HW.NetType	
Siemens.Engineering.HW.Enums.ProfinetUpdate-TimeMode		entfernt	
Siemens.Engineering.HW.Enums.RtClass	Byte	Siemens.Engineering.HW.RtClass	Int
Siemens.Engineering.HW.Enums.SignalDelaySe-lection	Byte	Siemens.Engineering.HW.SignalDelaySelection	Int
Siemens.Engineering.HW.Enums.SyncRole	Byte	Siemens.Engineering.HW.SyncRole	Int
Siemens.Engineering.HW.Enums.Transmission-RateAndDuplex	Uint	Siemens.Engineering.HW.TransmissionRateAnd-Duplex	Int

Änderungen für Attributwerte von Siemens.Engineering.HW.IoConnector

Attribut	Datentyp	Neuer Name	Datentyp
ProfinetUpdateTimeMode	ProfinetUpdateTimeMo-de	PnUpdateTimeAutoCalculation	Bool
ProfinetUpdateTime		PnUpdateTime	
AdaptUpdateTime		PnUpdateTimeAdaption	
WatchdogFactor		PNWatchdogFactor	
		DeviceNumber	String

Änderungen für Attributwerte von Siemens.Engineering.HW.IoController

Attribut	Datentyp	Neuer Name	Datentyp
		DeviceNumber	String

Änderungen für Attributwerte von Siemens.Engineering.HW.Node

Attribut	Datentyp	Neuer Name	Datentyp
HighestAddress		entfernt, nur am Subnetz verfügbar	
TransmissionSpeed		entfernt, nur am Subnetz verfügbar	
IsoProtocolUsed		UseIsoProtocol	
IpProtocolUsed		UseIpProtocol	
RouterAddressUsed		UseRouter	
PnDeviceNameAutoGenera-ted		PnDeviceNameAutoGeneration	
DeviceNumber		entfernt, zu IO-Connector / IO-Controller verschoben	

Änderungen für Attributwerte von Siemens.Engineering.HW.Subnet

Attribut	Datentyp	Neuer Name	Datentyp
HighestAddress	Byte	HighestAddress	Int
CableConfiguration		PbCableConfiguration	
RepeaterCount		PbRepeaterCount	
CopperCableLength		PbCopperCableLength	
OpticalComponentCount		PbOpticalComponentCount	
OpticalCableLength		PbOpticalCableLength	
OpticalRingEnabled		PbOpticalRing	
OlmP12		PbOlmP12	
OlmG12		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	Datentyp	Neuer Name	Datentyp
Tslot		PbTslot	
MinTsdr		PbMinTsdr	
MaxTsdr		PbMaxTsdr	
Tid1		PbTid1	
Tid2		PbTid2	
Trdy		PbTrdy	
Tset		PbTset	
Tqui		PbTqui	
Ttr		PbTtr	
TtrMs		entfernt	
TtrTypical		PbTtrTypical	
TtrTypicalMs		entfernt	
Watchdog		PbWatchdog	
WatchdogMs		entfernt	
Gap	Byte	PbGapFactor	Int
RetryLimit	Byte	PbRetryLimit	Int
IsochronMode		IsochronousMode	
AdditionalDevice		PbAdditionalPassivDeviceForIsochro-nousMode	
TotalDevice		PbTotalPassivDeviceForIsochronous-Mode	
DpCycleTimeAutoCalc		DpCycleMinTimeAutoCalculation	
TiToAutoCalc		IsochronousTiToAutoCalculation	
Ti		IsochronousTi	
To		IsochronousTo	

Änderungen für Attributwerte von Siemens.Engineering.Project

Attribut	Datentyp	Neuer Name	Datentyp
.HwExtensions		.HwUtilities	

Änderungen für Attributwerte von Siemens.Engineering.HW.Baudrate

Attribut	Datentyp	Neuer Name	Datentyp
BaudRate.BAUD_9600		BaudRate.BAUD9600	
BaudRate.BAUD_19200		BaudRate.BAUD19200	
BaudRate.BAUD_45450		BaudRate.BAUD45450	
BaudRate.BAUD_93750		BaudRate.BAUD93750	
BaudRate.BAUD_187500		BaudRate.BAUD187500	
BaudRate.BAUD_500000		BaudRate.BAUD500000	
BaudRate.BAUD_1500000		BaudRate.BAUD1500000	
BaudRate.BAUD_3000000		BaudRate.BAUD3000000	

Attribut	Datentyp	Neuer Name	Datentyp
BaudRate.BAUD_6000000		BaudRate.BAUD6000000	
BaudRate.BAUD_12000000		BaudRate.BAUD12000000	

Änderungen für Attributwerte von Siemens.Engineering.HW.CableLength

Attribut	Datentyp	Neuer Name	Datentyp
CableLength.Unknown		CableLength.None	
CableLength.Length_20m		CableLength.Length20m	
CableLength.Length_50m		CableLength.Length50m	
CableLength.Length_100m		CableLength.Length100m	
CableLength.Length_1000m		CableLength.Length1000m	
CableLength.Length_3000m		CableLength.Length3000m	

Änderungen für Attributwerte von Siemens.Engineering.HW.ChannelIoType

Attribut	Datentyp	Neuer Name	Datentyp
ChannelIoType.Unknown		ChannelIoType.Complex	

Änderungen für Attributwerte von Siemens.Engineering.HW.IpProtocolSelection

Attribut	Datentyp	Neuer Name	Datentyp
IpProtocolSelection.Address-Tailoring		IpProtocolSelection.ViaIoController	

Änderungen für Attributwerte von Siemens.Engineering.HW.TransmissionRateAndDuplex

Attribut	Daten-typ	Neuer Name	Datentyp
TransmissionRateAndDuplex.Unknown		TransmissionRateAndDuplex.None	
TransmissionRateAndDuplex.TP10Mbps_HalfDu-plex		TransmissionRateAndDuplex.TP10MbpsHalfDu-plex	
TransmissionRateAndDuplex.TP10Mbps_FullDu-plex		TransmissionRateAndDuplex.TP10MbpsFullDu-plex	
TransmissionRateAndDuplex.AsyncFi-ber10Mbps_HalfDuplex		TransmissionRateAndDuplex.AsyncFi-ber10MbpsHalfDuplex	
TransmissionRateAndDuplex.AsyncFi-ber10Mbps_FullDuplex		TransmissionRateAndDuplex.AsyncFi-ber10MbpsFullDuplex	
TransmissionRateAndDuplex.TP100Mbps_Half-Duplex		TransmissionRateAndDuplex.TP100MbpsHalf-Duplex	
TransmissionRateAndDuplex.TP100Mbps_FullDu-plex		TransmissionRateAndDuplex.TP100MbpsFullDu-plex	
TransmissionRateAndDuplex.FO100Mbps_Full-Duplex		TransmissionRateAndDuplex.FO100MbpsFull-Duplex	

Attribut	Daten-typ	Neuer Name	Datentyp
TransmissionRateAndDuplex.X1000Mbps_FullDu-plex		TransmissionRateAndDuplex.X1000MbpsFullDu-plex	
TransmissionRateAndDuplex.FO1000Mbps_Full-Duplex_LD		TransmissionRateAndDuplex.FO1000MbpsFull-DuplexLD	
TransmissionRateAndDuplex.FO1000Mbps_Full-Duplex		TransmissionRateAndDuplex.FO1000MbpsFull-Duplex	
TransmissionRateAndDuplex.TP1000Mbps_Full-Duplex		TransmissionRateAndDuplex.TP1000MbpsFull-Duplex	
TransmissionRateAndDuplex.FO10000Mbps_Full-Duplex		TransmissionRateAndDuplex.FO10000MbpsFull-Duplex	
TransmissionRateAndDuplex.FO100Mbps_Full-Duplex_LD		TransmissionRateAndDuplex.FO100MbpsFull-DuplexLD	
TransmissionRateAndDu-plex.POFPCF100Mbps_FullDuplex_LD		TransmissionRateAndDu-plex.POFPCF100MbpsFullDuplexLD	

9.2.4 Änderungen bei Export und Import

9.2.4.1 Änderungen bei Export und Import

Einleitung

Der Export und Import mittels TIA Portal Openness API wurde in V14 SP1 erweitert, um Kommentare bei Array-Elementen verarbeiten zu können. Dadurch wurde ein neues Schema erforderlich. Für den Import und Export von Bausteinschnittstellen werden ab jetzt zwei Schemaversionen verwendet:

- Für den Import: Die Entscheidung über die verwendete Schemaversion wird anhand des Namensraums getroffen: <Sections xmlns=http://www.siemens.com/automation/Openness/SW/Interface/v2>
- Für den Export: Die Entscheidung über die verwendete Schemaversion wird anhand der Projektversion getroffen. Projekte V14 SP1 führen zu Version 2, Projekte V14 führen zu Version 1.

9.2.4.2 Änderungen in der API

Quelle generieren

Die folgenden Methoden wurden aus ProgramBlocks entfernt:

- GenerateSourceFromBlocks
- GenerateSourceFromTypes

Die folgenden Methoden wurden ergänzt:

- GenerateSource bis PlcExternalSourceSystemGroup

Beispiel

```
// 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);
```

9.2.4.3 Schemaerweiterung

Schemaerweiterung für Kommentare und Startwerte

Kommentare und Startwerte werden in dem neuen Element "Subelement" gespeichert, das sich auf das Array-Element mit dem Attribut "Path" bezieht.

Subelement enthält den Startwert und den Kommentar für das referenzierte Array-Element. Das Attribut "Path" an StartValue wird im neuen Schema entfernt.

Schemadefinition von "Subelement":

```
<xs:element name="Subelement" type="Subelement_T"/>
  <xs:complexType name="Subelement_T">
      <xs:sequence>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element ref="StartValue"/>
              <xs:element ref="Comment"/>
          </xs:choice>
      </xs:sequence>
      <xs:attribute name="Path" type="IndexPath_TP"/>
  </xs:complexType>
```

Erweiterung des Mitgliedstyps:

```
<xs:complexType name="Member_T">
       <xs:sequence>
           <xs:element ref="AttributeList" minOccurs="0" maxOccurs="1"/>
           <xs:choice minOccurs="0" maxOccurs="unbounded">
               <xs:element ref="Member"/>
               <xs:element ref="Sections"/>
               <xs:element ref="StartValue"/>
               <xs:element ref="Comment"/>
               <xs:element ref="Subelement"/>
           </xs:choice>
       </xs:sequence>
</xs:complexType>
```

Beispiele:

Speicherung von Kommentaren und Startwerten in einfachen Arrays:

```
<Member Name="Static_1" Datatype="Array[0..1] of Bool">
    <Comment>
        <MultiLanguageText Lang="de-DE">comment for array</MultiLanguageText>
    </Comment>
    <Subelement Path="0">
        <StartValue>true</StartValue>
        <Comment>
            <MultiLanguageText Lang="de-DE">comment for array element 0</MultiLanguageText>
        </Comment>
    </Subelement>
    <Subelement Path="1">
        <StartValue>true</StartValue>
        <Comment>
            <MultiLanguageText Lang="de-DE">comment for array element 1</MultiLanguageText>
        </Comment>
    </Subelement>
</Member>
```

Speicherung von Kommentaren und Startwerten in Arrays aus UDT:

```
<Member Name="Static_1" Datatype="Array[0..1] of &quot;User_data_type_1&quot;">
    <Comment>
        <MultiLanguageText Lang="de-DE">comment for array</MultiLanguageText>
    </Comment>
    <Subelement Path="0">
        <Comment>
            <MultiLanguageText Lang="de-DE">cmt array  0</MultiLanguageText>
        </Comment>
    </Subelement>
    <Sections>
        <Section Name="None">
            <Member Name="Element_1" Datatype="Bool">
                <Subelement Path="0">
                    <StartValue>true</StartValue>
                    <Comment>
                        <MultiLanguageText Lang="de-DE">comment for element 0</MultiLanguageText>
                    </Comment>
                </Subelement>
                <Subelement Path="1">
                    <StartValue>true</StartValue>
                    <Comment>
                        <MultiLanguageText Lang="de-DE">comment for element 1</MultiLanguageText>
                    </Comment>
                </Subelement>
            </Member>
            <Member Name="Element_2" Datatype="Struct">
                <Member Name="Element_1" Datatype="Int">
                    <Subelement Path="0">
                        <StartValue>11</StartValue>
                        <Comment>
                            <MultiLanguageText Lang="de-DE">comment for element 0</MultiLanguageText>
                        </Comment>
                    </Subelement>
                </Member>
            </Member>
        </Section>
    </Sections>
</Member>
```

Speicherung von Kommentaren und Startwerten in Arrays aus Struct:

```
<Member Name="Static_1" Datatype="Array[0..1] of Struct">
    <Member Name="Static_1" Datatype="Int">
        <Subelement Path="0">
            <StartValue>11</StartValue>
            <Comment>
                <MultiLanguageText Lang="de-DE">comment for int elem</MultiLanguageText>
            </Comment>
        </Subelement>
    </Member>
    <Member Name="Static_2" Datatype="Bool">
        <Subelement Path="1">
            <StartValue>true</StartValue>
            <Comment>
                <MultiLanguageText Lang="de-DE">comment for bool elem</MultiLanguageText>
            </Comment>
        </Subelement>
    </Member>
</Member>
```

9.2.4.4 Schemaänderungen

Knoten Access in SW.PlcBlocks.Access.xsd

Das Attribut Type des Knotens Access wurde zu den untergeordneten Knoten von Access verschoben bei

- AbsoluteOffset – erforderlich
- Address – optional

```
<StlStatement UId="22">
  <StlToken Text="L" />
  <Access Scope="Address">
    <Address Area="Local" Type="Word" BitOffset="80" />
  </Access>
</StlStatement>
```

Das Attribut Type von Constant wurde mit dem neuen Unterknoten ConstantType ersetzt.

```
<Access Scope="LocalConstant">
  <IntegerAttribute Name="NumBLs" Informative="true">5</IntegerAttribute>
  <Constant Name="LocalConstant_A">
    <ConstantType Informative="true">Int</ConstantType>
    <ConstantValue Informative="true">10</ConstantValue>
    <StringAttribute Name="Format" Informative="true">Dec_signed</StringAttribute>
  </Constant>
</Access>
```

Der Wert des Attributs Scope in Access wurde in TypedConstant umbenannt, wenn der ConstantValue einen typqualifizierten Wert enthält (z. B.: int#10).

Constant hat kein Attribut Type, wenn ConstantValue einen typqualifizierten Wert enthält (z. B.: int#10).

Lokale Variablen haben keinen Knoten Address, wenn Scope eine LocalVariable ist.

Wenn ein Access auf beliebiger Ebene in einem anderen Access verschachtelt ist, muss nur der äußere Access eine UId haben.

Knoten Address in SW.PlcBlocks.Access.xsd

Das Attribut BitOffset des Knotens Address ist optional geworden.

Die Deklarationen zum Exportieren des absoluten Zugriffs wurde wie in der folgenden Tabelle gezeigt geändert:

Bereich ab V14 SP1	Typ	Bausteinnummer	Bit-Offset	Beispiel
DB	Block_DB	obligatorisch	verboten	OPN %DB12
DB	unsortiert	vorhanden	obligatorisch	%DB100.DBX10.3
DB	unsortiert	nicht vorhanden	obligatorisch	%DB100.DBX10.3
L	unsortiert	verboten	obligatorisch	%LW10.0

Bereich ab V14 SP1	Typ	Bausteinnummer	Bit-Offset	Beispiel
I Q M	unsortiert	verboten	obligatorisch	%I0.0 %Q0.0 %M0.0
T C	unsortiert	verboten	obligatorisch	%T0 %C1
Block_FC Block_FC	Block_FC Block_FB	obligatorisch	verboten	Aufruf %FB4, %DB5 Input_1 := %FC10 Aufruf %FB4, %DB5 Input_2 := %FB11
Peripherieeingang	unsortiert	verboten	obligatorisch	
Peripherieausgang	unsortiert	verboten	obligatorisch	

Knoten Area in SW.PlcBlocks.Access.xsd

Der Knoten Area hat eine vereinfachte Enum-Liste erhalten:

- LocalC und LocalN wurden zu Local
- DBc, DBv, DBr wurden entfernt.

Knoten CallInfo in SW.PlcBlocks.Access.xsd

Das Attribut Name des Knotens CallInfo ist optional geworden.

Das Attribut BlockType des Knotens CallInfo ist erforderlich geworden.

+2.2.5 Anwenderbausteinaufrufe

Knoten Constant in SW.PlcBlocks.Access.xsd

Der Knoten Constant referenziert den Knoten CostantType mit minOccurs=0.

Der Knoten Constant referenziert den Knoten IntegerAttribute nicht mehr.

Knoten ConstantValue in SW.PlcBlocks.Access.xsd

Der Knoten ConstantValue erhält ein informatives Attribut.

Knoten Instruction in SW.PlcBlocks.Access.xsd

Der Knoten Instruction referenziert den Knoten Acces mit minOccurs=0.

Die Parameterattribute Section, Type und TemplateReference wurden bei Instruction gelöscht.

Knoten Parameter in SW.PlcBlocks.Access.xsd

Das Attribut SectionName des Knotens Parameter ist optional geworden.

Werte für Scope in SW.PlcBlocks.Access.xsd

Die Enum-Liste von Scope wurde erweitert um:

- TypedConstant
- AddressConstant
- LiteralConstant
- AlarmConstant
- Address
- Statusword
- Expression
- Call
- CallWithType

Knoten Statusword in SW.PlcBlocks.Access.xsd

Die Enum-Liste von Statusword wurde erweitert um:

- STW

Knoten ConstantType in SW.PlcBlocks.Access.xsd

Der neue Knoten ConstantType wurde mit dem optional verwendeten Attribut Informative eingeführt.

Knoten CallRef in SW.PlcBlocks.LADFBD.xsd

Der Knoten CallRef wurde in Call umbenannt und hat keinen Unterknoten BooleanAttribute mehr.

Knoten InstructionRef in SW.PlcBlocks.LADFBD.xsd

Der Knoten InstructionRef wurde durch den Knoten Part ersetzt.

Knoten Part in SW.PlcBlocks.LADFBD.xsd

Der neue Knoten ConstantType wurde eingeführt und ersetzt den Knoten InstructionRef.

- Attribute: Name und Version
- Unterknoten: Der Unterknoten Instruction als neue Wahl neben dem vorhandenen Equation
- hat weder einen Unterknoten BooleanAttribute noch ein Attribut Gate.

Knoten Wire in SW.PlcBlocks.LADFBD.xsd

Das Attribut Name des Knotens Wire wurde entfernt.

Knoten TemplateReference in SW.PlcBlocks.LADFBD.xsd

Der Knoten TemplateReference wurde gelöscht.

Knoten StatementList in SW.PlcBlocks.STL.xsd

Enum-Liste von StatementList (STL_TE):

- L_STW wurde entfernt
- T_STW wurde entfernt

9.2.4.5 Verhaltensänderungen

Absoluter Zugriff

In V14 wurde der Import des absoluten Zugriffs für die meisten Kombinationen abgebrochen. Ab V14 SP1 funktioniert der Import des absoluten Zugriffs für die folgenden Bereiche:

- Eingang
- Ausgang
- Merker
- Zeit, sofern auf dem PLC unterstützt
- Zähler, sofern auf dem PLC unterstützt
- DB
- DI

Wenn ein symbolischer Zugriff und ein absoluter Zugriff gleichzeitig verwendet und nicht von der Prüfung des Schemas oder der Knotenart abgelehnt werden, ist der Import nur dann erfolgreich, wenn die Boxzugriffsinformationen erfolgreich aufgelöst werden. Wenn der symbolische Zugriff zu anderen Informationen führt als der absolute Zugriff, wird der Import abgelehnt.

```
<Access Scope="Address">
    <Address Area="Memory" Type="Word" BitOffset="0" />
</Access>


<CallInfo Name="Block_1" BlockType="FC">
    <Parameter Name="Input_1" Section="Input" Type="Int" />
    <Parameter Name="Input_1" Section="Input" Type="Int" />
<!-- Import will be aborted because parameter name 'Input_1' is used more than once -->
    <Parameter Name="Output_1" Section="NotExisting" Type="Int" />
<!-- Import will be aborted because the section 'NotExisting' can not be used. -->
    <Parameter Name="ENO" Section="Output" Type="Int" />
<!-- Import will be aborted because the parameter name 'ENO' is restricted and can not be used. -->
</CallInfo>
```

Direkter DB-Zugriff

Ab V14 SP1 kann der indirekte DB-Zugriff nur dann importiert werden, wenn 'Offset', 'Typ' und 'Symbol' angegeben werden.

```
...
    <Access Scope="LocalVariable" UId="21">
        <Symbol>
            <Component Name="Output_3" />
            <AbsoluteOffset BitOffset="16" Type="Word" />
        </Symbol>
    </Access>
...
```

Symbolische und absolute Informationen für lokalen Zugriff

Beim Importieren von "symbolischem Zugriff" werden alle möglichen angegebenen "Informationen für absoluten Zugriff" überprüft, sofern sie nicht als "informativ" gekennzeichnet sind. Ab V14 SP1 wird der Import abgebrochen, wenn die absoluten Informationen nicht übereinstimmen.

Einschränkungen der Bausteinschnittstelle

In V14SP1 werden verschiedene Einschränkungen geprüft. Diese Einschränkungen sind Benutzern des Bausteinschnittstellen-Editors wohlbekannt. Immer wenn der Bausteinschnittstellen-Editor einen Parameter durch Ergänzen oder Erhöhen von '_1' umbenennt, wird der OPNS-Import abgebrochen.

Beispielsweise werden die folgenden Einschränkungen überprüft:

- Doppelte Parameternamen
- Falsche Abschnittsnamen, einschließlich 'Return-Abschnitt' bei FB-Bausteinen
- Eingeschränkte Wörter

Sortierung der Abschnitte beim Import

Wenn der aufgerufene Baustein zum Zeitpunkt des Imports nicht vorhanden ist, wird die Schnittstellendefinition auf Aufrufseite dazu verwendet, den aufgerufenen Anwenderbaustein anzuzeigen. In V14 SP1 werden die Abschnitte in der Reihenfolge sortiert, in der sie in der Bausteinschnittstelle des aufgerufenen Bausteins angezeigt werden würden, wenn dieser mit den gleichen Parametern vorhanden wäre.

Die Abschnittsreihenfolge der importierten Parameter lautet:

- Eingang
- Ausgang

Das folgende Beispiel in AWL xml

```
<StlStatement UId="21">
<StlToken Text="CALL" />
    <Access Scope="Call">
        <CallInfo Name="Block_2" BlockType="FC">
            <Parameter Name="Output_1" Section="Output" Type="Int">
                <Access Scope="GlobalVariable">
                    <Symbol>
                        <Component Name="Tag_3" />
                    </Symbol>
                </Access>
            </Parameter>
            <Parameter Name="Input_1" Section="Input" Type="Int">
                <Access Scope="GlobalVariable">
                    <Symbol>
                        <Component Name="Tag_1" />
                    </Symbol>
                </Access>
            </Parameter>
            <Parameter Name="Output_2" Section="Output" Type="Int">
                <Access Scope="GlobalVariable">
                    <Symbol>
                        <Component Name="Tag_4" />
                    </Symbol>
                </Access>
            </Parameter>
            <Parameter Name="Input_2" Section="Input" Type="Int">
                <Access Scope="GlobalVariable">
                    <Symbol>
                        <Component Name="Tag_2" />
                    </Symbol>
                </Access>
            </Parameter>
        </CallInfo>
    </Access>
</StlStatement>
```

führt zu:

```
CALL  "Block_2"
    Input_1  :="Tag_1"
    Input_2  :="Tag_2"
    Output_1 :="Tag_3"
    Output_2 :="Tag_4"
```

Eindeutige Aufrufernamen von Anwenderbausteinen

Im TIA Portal müssen Namen eindeutig sein. Das bedeutet beispielsweise, dass eine Variable nicht den gleichen Namen wie ein Baustein haben kann. Für den XML-Import der TIA Portal Openness API bedeutet dies für den Fall, dass die XML einen Anwenderbausteinaufruf enthält, bei dem der aufgerufene Baustein zum Zeitpunkt des Imports nicht vorhanden ist, der Name des aufgerufenen Bausteins verglichen mit allen vorhandenen Namen im Projekt eindeutig sein muss. Ist der Name des aufgerufenen Bausteins nicht eindeutig, wird der Import abgebrochen.

Im folgenden Beispiel wird der Import abgebrochen, weil der Name des aufgerufenen Bausteins "Tag_1" bereits für eine Variablentabelle verwendet wird.

```
...
    <SW.Tags.PlcTag ID="1" CompositionName="Tags">
        <AttributeList>
            <DataTypeName>Int</DataTypeName>
            <LogicalAddress>%MW2</LogicalAddress>
            <Name>Tag_1</Name>
        </AttributeList>
    </SW.Tags.PlcTag>
...
...
    <StlStatement UId="21">
        <StlToken Text="CALL" />
        <Access Scope="Call">
            <CallInfo Name="Tag_1" BlockType="FC">
                <Parameter Name="Input_1" Section="Input" Type="Int">
                    <Access Scope="GlobalVariable">
                        <Symbol>
                            <Component Name="Tag_1" />
                        </Symbol>
                    </Access>
                </Parameter>
...
```

Im folgenden Beispiel wird der Import abgebrochen, weil zwei Parameter den gleichen Namen "Input1" haben.

```
<StlStatement UId="22">
    <StlToken Text="CALL" />
    <Access Scope="Call">
        <CallInfo Name="Block_1" BlockType="FB">
            <Instance Scope="GlobalVariable">
                <Component Name="Block_1_DB" />
            </Instance>
            <Parameter Name="Input1" Section="Input" Type="Int">
                <Access Scope="GlobalVariable">
                    <Symbol>
                        <Component Name="Tag_9" />
                    </Symbol>
                </Access>
            </Parameter>
            <Parameter Name="Input1" Section="Input" Type="Time">
                <Access Scope="TypedConstant">
                    <Constant>
                        <ConstantValue>T#1s</ConstantValue>
                    </Constant>
                </Access>
            </Parameter>
        </CallInfo>
    </Access>
</StlStatement>
```

Bibliotheksbausteinaufrufe

Die importierte XML kann Aufrufe von Anwenderbausteinen enthalten. Diese Anwenderbausteine werden anhand des Namens identifiziert.

Anwenderbausteine können auch Bibliothekselemente aufrufen. Diese Bibliothekselemente können als 'Bibliotheksbausteinaufrufe' generiert werden. Da Bibliotheksbausteinen denselben Namensraum wie Anwenderbausteine verwenden, kann bei einem anhand des

Namens durchgeführten Imports eines Anwenderbausteinaufrufs die Implementierung eines Anwenderbausteins aufgerufen werden.

Vor V14 SP1 hat der Importvorgang versucht, die Parameter zwischen dem Anwenderbausteinaufruf und dem Anweisungsbausteinaufruf zuzuordnen. Gelegentlich wurde der Import abgebrochen, gelegentlich wurden beim Import alle nicht übereinstimmenden Parameter gelöscht.

Ab V14 SP1 findet der Anwenderbausteinaufruf immer noch den Bibliotheksbaustein, doch der Aufruf wird nicht ungültig.

Nichtübereinstimmung des Bausteintyps

Wenn die XML einen Anwenderbausteinaufruf von 'Block_1' mit mehr Parametern als die entsprechende FC im Projekt enthält, definiert der Import ab V14 SP1 eine neue Schnittstelle für den aufgerufene Baustein, die dem Anwenderbausteinaufruf aus der XML entspricht. Bei der nächsten Übersetzung des Programmbausteins wird versucht, den Aufruf zu aktualisieren.

Neue Funktionsumfänge für Konstanten

Ab V14SP1 wurden verschiedene neue Funktionsumfänge für Konstanten definiert. Der Import ist nur dann erfolgreich, wenn die Werte in der XML-Datei dem Funktionsumfang der Konstante entsprechen. Der Import wird möglicherweise abgebrochen, wenn nicht alle für eine Konstante angegebenen Informationen der vorhandenen Konstante entsprechen.

```
...
    <Access Scope="LiteralConstant">
        <Constant>
            <ConstantType>Int</ConstantType>
            <ConstantValue>16#0000_0001</ConstantValue>
        </Constant>
    </Access>
...
    <Access Scope="TypedConstant">
        <Constant>
            <ConstantValue>Int#10</ConstantValue>
        </Constant>
    </Access>
...
    <Access Scope="LiteralConstant">
        <Constant>
            <ConstantType>Int</ConstantType>
            <ConstantValue>10</ConstantValue>
        </Constant>
    </Access>
...
    <Access Scope="GlobalConstant">
        <Constant Name="Constant_1" />
    </Access>
...
    <Access Scope="LocalConstant">
        <Constant Name="Constant_1" />
    </Access>
...
    <Access Scope="AddressConstant">
        <Constant Name="Tag_1" />
    </Access>
...
    <Access Scope="AlarmConstant">
        <Constant>
            <ConstantType>C_Alarm</ConstantType>
            <ConstantValue>16#0000_0001</ConstantValue>
        </Constant>
    </Access>
...
```

Anmerkungen von Anweisungsversionen

Ab V14 SP1 können nur Anweisungsversionen importiert werden, die auf dem PLC einsetzbar sind, in den importiert werden soll. Ist in der XML keine Anweisungsversion angemerkt, wird die im PLC ausgewählte Version verwendet. In KOP und FUP gibt es für einige als Anweisungen dargestellte Elemente keine Versionierung. Diese Elemente können nur ohne Version importiert werden.

```
...
    <Part Name="MIN" Version="1.0" UId="27" DisabledENO="false">
        <TemplateValue Name="card" Type="Cardinality">2</TemplateValue>
        <TemplateValue Name="value_type" Type="Type">Int</TemplateValue>
    </Part>
    <Part Name="MIN" UId="28" DisabledENO="false">
        <TemplateValue Name="card" Type="Cardinality">2</TemplateValue>
        <TemplateValue Name="value_type" Type="Type">Int</TemplateValue>
    </Part>
...
```

Deaktivierter ENO

Die Funktion zum Deaktivieren des ENO dient auf S7-1200 und S7-1500 PLCs dazu, Laufzeit verbrauchende Berechnungen des ENO-Verbindungszustand zu deaktivieren.

Ab V14 SP1 kann der Merker DisabledENO nur auf PLCs importiert werden, die diese Funktion unterstützen.

```
...
    <Part Name="Add" UId="24" DisabledENO="false">
        <TemplateValue Name="Card" Type="Cardinality">2</TemplateValue>
        <TemplateValue Name="SrcType" Type="Type">Int</TemplateValue>
    </Part>
...
```

Typvalidierung für absoluten Zugriff auf den L-Stack

Ab V14 SP1 wird der Import abgebrochen, wenn der Typ nicht verwendet oder zugeordnet werden kann.

Validierung von Index-Idents

Indexzugriff ist verwendbar, wenn die Funktion "Symbolischer Zugriff auf den Speicher" definiert ist, beispielsweise lokaler Zugriff, globaler Zugriff, indirekter Zugriff.

Wenn eine literale Konstante als Index verwendet wird, werden die Typen Ganzzahl mit und ohne Vorzeichen in Dint geändert. Ab V14 SP1 wird der Import abgebrochen, wenn ein Typ außerhalb des angegebenen Bereichs angegeben wird.

Sämtlicher Indexzugriff wird daraufhin geprüft, ob die Art des Zugriffs überhaupt als 'Indexzugriff' verwendet werden kann. Ab V14 SP1 wird der Import abgebrochen, wenn der definierte Indexzugriff nicht verwendet werden kann.

Sortierung der Elementreihenfolge

Ab V14 SP1 werden die Elemente in KOP und FUP in der Reihenfolge der Codeerzeugung sortiert, wo beim Export automatisch möglich. In einigen seltenen Fällen kann die exportierte XML nicht wieder importiert werden. In diesen Fällen muss entweder die XML angepasst werden oder die entsprechenden Netzwerke müssen gelöscht und neu programmiert werden. Doch die Reihenfolge von Leitern und Referenzen ist immer noch nicht zuverlässig.

Alarmkonstanten

Mit V14 SP1 wurden die Übersetzungsprüfungen auf gültige Alarmkonstanten erweitert. Es kann vorkommen, dass aufgrund einer in V14 mit defekten Alarmkonstanten importierten XML ein Projekt in V14 SP1 übersetzbar ist. Öffnen Sie in diesem Fall das relevante Netzwerk im KOP/FUP-Editor und löschen Sie den tatsächlichen Alarmoperanden. Der Editor erstellt automatisch eine neue gültige Alarmkonstante.

```
<FlgNet>
    <Parts>
        <Access Scope="AlarmConstant" UId="21">
            <Constant>
                <ConstantType>C_Alarm</ConstantType>
                <ConstantValue>16#0000_0002</ConstantValue>
            </Constant>
        </Access>
        <Call UId="22">
            <CallInfo Name="Block_1" BlockType="FB">
                <Instance UId="23" Scope="GlobalVariable">
                    <Component Name="Block_1_DB" />
                </Instance>
                <Parameter Name="Input_1" Section="Input" Type="C_Alarm" />
            </CallInfo>
        </Call>
    </Parts>
    <Wires>
        <Wire UId="24">
            <Powerrail />
            <NameCon UId="22" Name="en" />
        </Wire>
        <Wire UId="25">
            <IdentCon UId="21" />
            <NameCon UId="22" Name="Input_1" />
        </Wire>
    </Wires>
</FlgNet>
```

Einschränkungen für Instanzen von Anwenderbausteinen und Anweisungen

In V14 war es möglich, FC-Anwenderbausteinaufrufe mit einer Instanz zu importieren und diese Aufrufe gelegentlich sogar zu übersetzen.

Ab V14 SP1 ist der Import von Instanzen nur möglich, wo Instanzen unterstützt werden. Bestehende Projekte mit Instanzen an FC-Anwenderbausteinaufrufen und Anweisungen lassen sich möglicherweise nicht mehr übersetzen. In diesem Fall muss der Aufruf gelöscht und neu programmiert werden. Jeder Versuch, eine Aufrufaktualisierung oder eine andere Art der automatischen Reparatur durchzuführen, schlägt fehl.

EnEno sichtbar

In V14 waren die EN- und ENO-Anschlüsse von 'InstructionRef' abhängig vom Merker ENENO verwendbar oder nicht.

Ab V14SP1 werden vom OPNS während des Imports basierend auf dem Element und der Verdrahtung entweder die EN- oder die ENO-Anschlüsse verwendet. Aufgrund dieser automatischen Erkennung ist eine unterschiedliche Verwendung von EN- und ENO-Anschlüssen festzustellen. Höchstwahrscheinlich zeigen nur die Boxen von IEC-Zeiten und IEC-Zählern gewisse Probleme.

UId-Zuweisung

Die Zuweisung von UIds zu Teilen, Zugriffen und Leitern ändert sich mit V14 SP1. Die UIds für Ausdrücke, CallInfo und Operanden müssen in einer Übersetzungseinheit eindeutig sein. Aus TIA Portal-Perspektive sind die UIds in der XML Schlüssel ohne zusätzliche Bedeutung neben der Identifikation eines Elements.

Prüfen von Zeichenfolgen

Strengere Prüfungen bezüglich Anführungszeichen, Stellvertreterzeichen und Steuerzeichen werden für das Attribut Name durchgeführt beim Import von

- IntegerAttribute
- StringAttribute
- DateAttribute
- AutomaticTyped
- Component
- Invisible
- Label
- NameCon
- Negated
- TemplateValue
- CallInfo
- Instruction
- Parameter
- Part
- Step

Strengere Prüfungen bezüglich Stellvertreterzeichen und Steuerzeichen werden durchgeführt beim Import von

- Bausteintiteln und Netzwerken
- LineComment-Text
- Konstanten Zeichenfolgen (Typen String, WString, Char, Wchar)

Strengere Prüfungen bezüglich Stellvertreterzeichen und Steuerzeichen (Tab und neue Zeile zulässig) werden durchgeführt beim Import von

- Kommentaren und Netzwerken
- String-Attributen
- Knoten, die mehrsprachige Texte definieren wie Alarmtext, Comments
- Token-Texte

Nichtbeachtung von Groß-/Kleinschreibung bei Vorlagenoperationen und Parametern

Ab V14 SP1 wird die Nichtbeachtung von Vorlagenoperationen für Anweisungen und Aufruf- oder Anweisungsparameter importiert und automatisch korrigiert.

Der folgende Code wird importiert und der fehlerhafte Wert "Eq" wird in "EQ" korrigiert und der fehlerhafte Parameter "iN1" wird in "IN1" korrigiert:

```
<StlStatement 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>
```

In Aufrufen verwendete Multiinstanzen

Ab V14 SP1 wird der Import abgebrochen, wenn die in einem Aufruf verwendete Multiinstanz nicht vorhanden ist.

Der folgende Code zeigt ein XML-Beispiel, in dem die Multiinstanz im Schnittstellenabschnitt korrekt definiert ist:

```
<SW.Blocks.FB ID="0">
    <AttributeList>
        <Interface>
            <Sections xmlns="http://www.siemens.com/automation/Openness/SW/Interface/v2">
                <Section Name="Input" />
                <Section Name="Output" />
                <Section Name="InOut" />
                <Section Name="Static">
                    <!-- The next line must be present if multiinstance is used in code-->
                    <Member Name="Static_1" Datatype="&amp;quot;Block_2&amp;quot;" />
                </Section>
    ....
    <StlStatement UId="22">
        <StlToken Text="CALL" />
        <Access Scope="Call">
            <CallInfo BlockType="FB">
                <!-- Multiinstace usage-->
                <Instance Scope="LocalVariable">
                    <Component Name="Static_1" />
                </Instance>
                <Parameter Name="Input_1" Section="Input" Type="Int">
                    <Access Scope="GlobalVariable">
                        <Symbol>
                            <Component Name="Tag_9" />
                        </Symbol>
                    </Access>
                </Parameter>
            </CallInfo>
        </Access>
    </StlStatement>
```

Vorlagenkardinalitäten in AWL

In AWL haben die Vorlagenkardinalitäten für jede Anweisung einen festen Standardwert, der der einzig gültige Wert ist. Ab V14 SP1 wird der Import abgebrochen, wenn ein anderer Wert für die Kardinalität verwendet wird.

Indirekten Zugriff importieren

Ab V14 SP1 kann der indirekte Zugriff nur dort importiert werden, wo er übersetzt werden kann.

```
<StlStatement UId="22">
    <StlToken Text="L" />
    <Access Scope="Address">
        <Indirect Width="Word" Area="Memory">
            <Access Scope="LocalVariable">
                <Symbol>
                    <Component Name="Temp_1" />
                </Symbol>
            </Access>
        </Indirect>
    </Access>
</StlStatement>
```

Statuswörter Importieren

Ab V14 SP1 kann das Statuswort nur bei Anweisungen importiert werden, bei denen es unterstützt wird.

- L - Unterstütztes Statuswort: STW
- T - Unterstütztes Statuswort: STW
- A - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- AN - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- O - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- ON - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- X - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU
- XN - Unterstütztes Statuswort: BR, OV, OS, EQ, NE, GT, Lt, GE, LE, U0, NU

Hinweis

Die meisten Statuswörter sind nur bei S7-300 und S7-400 PLCs nützlich.

Leere Anweisungen

Der Import wird abgebrochen, wenn eine Anweisung keinen Knoten <StlStatement/> hat. Bei einer leeren Anweisung ergänzen Sie den Knoten <StlToken Text="Empty_Line" />.

Der Import wird abgebrochen, wenn eine leere Anweisung über Kommentare verfügt. Bei einer Anweisung nur mit Kommentaren verwenden Sie <StlToken Text="COMMENT" />.

```
<!-- Declaration of an empty statement -->
<StlStatement UId="23">
    <StlToken Text="EMPTY_LINE" />
</StlStatement>

<!-- Declaration of a statement with only comments-->
<StlStatement UId="22">
    <LineComment>
        <Text>Comment number 1</Text>
    </LineComment>
    <StlToken Text="COMMENT" />
</StlStatement>
```

9.2.4.6 Änderungen an Bausteinattributen

Änderungen an allgemeinen Attributen

AutoNumber hat einen neuen Standardwert (false) bei klassischen OBs.

HeaderVersion hat einen neuen Typ System.Version (anstelle von String).

IsKnowHowProtected wird auch auf benutzerdefinierte Datentypen angewendet.

ILibraryTypeInstance.ConnectedVersion, ILibraryTypeInstance.Dependencies, ILibraryTypeInstance.Dependents wurden aus der Tabelle der allgemeinen Attribute entfernt, weil sie weder in XML exportiert werden noch über API zugänglich sind.

MemoryLayout hat einen neuen Standardwert: Standard bei klassischen PLCs und Optimized bei Plus-PLCs.

Number wird auch auf benutzerdefinierte Datentypen angewendet, und es wird in XML dargestellt und ist auch über API zugänglich.

Änderungen an spezifischen Attributen

IsOnlyStoredInLoadMemory und IsWriteProtectedInAS sind jetzt für IDBofUDT schreibgeschützt, sofern der UDT zu einem Systembibliothekselement gehört.

OfSystemLibElement und OfSystemLibVersion gehören jetzt nicht mehr zu den allgemeinen, sondern zu den spezifischen Attributen.

OfSystemLibVersion hat einen neuen Typ System.Version (anstelle von String).

ParameterPassing bleibt Lesen/Schreiben bei FCs und FBs nur, wenn

- ProgrammingLanguage AWL ist und
- MemoryLayout Standard ist und
- die Schnittstelle leer ist

GraphVersion hat einen neuen Typ System.Version (anstelle von String).

Ein neues Attribut namens ExtensionBlockName wird für in Graph geschriebene FBs (ab Graph Version V4) eingeführt.

Ein neues Attribut namens InvalidValuesAcquisition wird für in Graph geschriebene FBs (ab Graph Version V4) eingeführt.

Ein neues Attribut namens IsWriteProtected wird für Codebausteine eingeführt.

DownloadWithoutReinit ist jetzt schreibgeschützt und gilt auch für IDBofFBs.

Supervisions ist jetzt bei IDBofFBs schreibgeschützt.

Änderungen in Enums

Die Enum-Werte für ProgrammingLanguage wurden wie folgt geändert:

- Ein neuer Enum-Wert F_CALL wurde eingeführt.
- Ein neuer Enum-Wert Motion_DB wurde für das Technologieobjekt Motion eingeführt.
- GRAPH_SEQUENCE, GRAPH_ACTIONS, GRAPH_ADDINFOS wurden aus den Enums gelöscht. Sie wurden durch GRAPH ersetzt.

Die Enum-Werte für BlockType wurden wie folgt geändert:

- Die Werte OB, FC, DB, SFC wurden gelöscht, weil diese Enums lediglich beim Attribut InstanceOfType verwendet werden.

9.3 Die wichtigsten Änderungen in V14

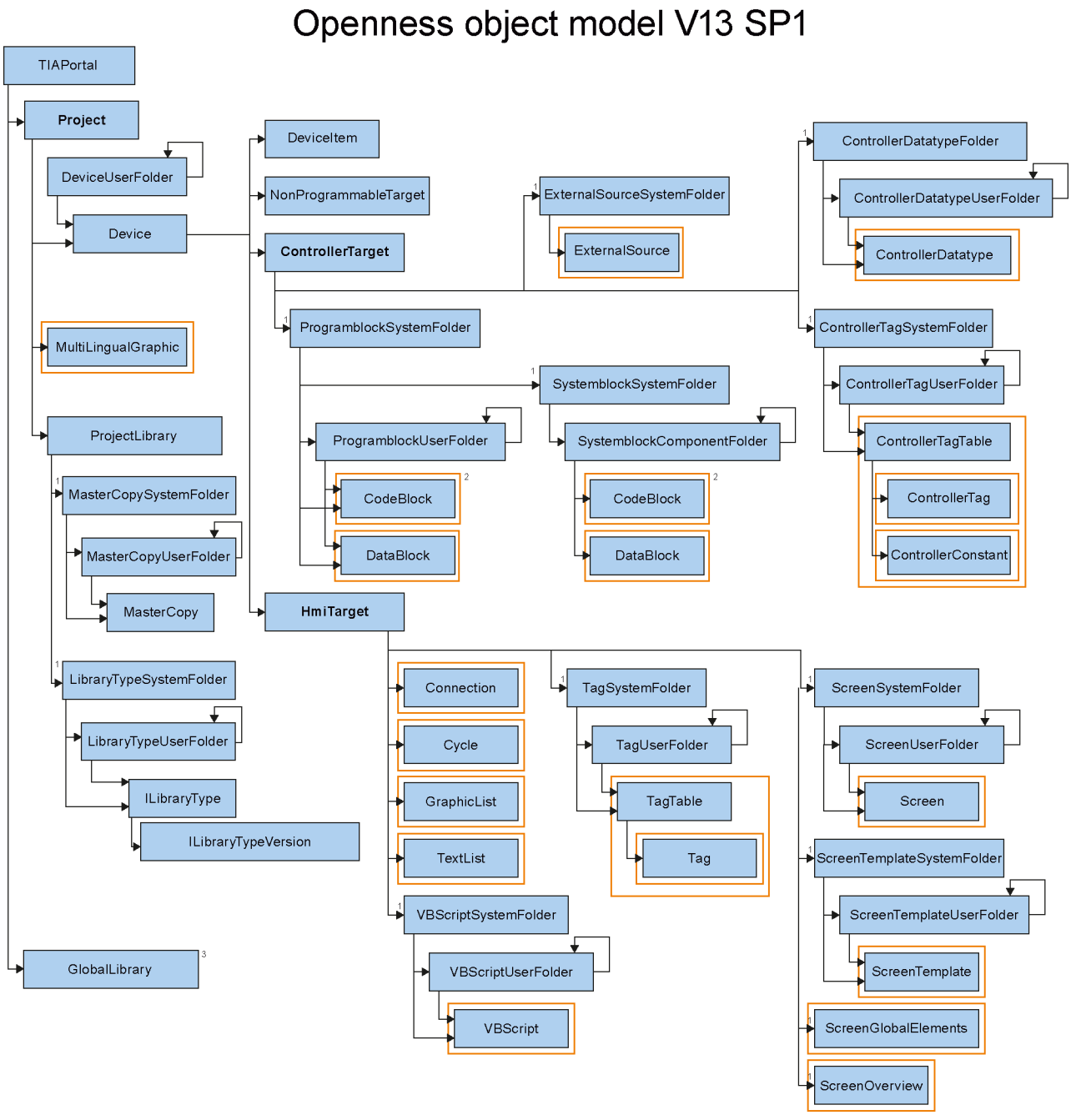
9.3.1 Wesentliche Änderungen des Objektmodells

Objektmodell von TIA Portal Openness V13 SP1 und älter

Um Ihnen einen Vergleich zwischen dem alten und dem neuen Objektmodell von TIA Portal Openness zu ermöglichen, beschreibt das nachstehende Diagramm das Objektmodell von TIA Portal V13 SP1.

Hinweis

Das in dem Diagramm beschriebene Objektmodell ist veraltet. Sie finden Informationen über das Objektmodell von TIA Portal Openness V14 SP1 unter TIA Portal Openness-Objektmodell (Seite 51)


Openness object model V13 SP1
TIAPortal
Project
DeviceUserFolder
Device
DeviceItem
NonProgrammableTarget
ControllerTarget
ExternalSourceSystemFolder
ExternalSource
ControllerDatatypeFolder
ControllerDatatypeUserFolder
ControllerDatatype
MultiLingualGraphic
ProgramblockSystemFolder
ControllerTagSystemFolder
ControllerTagUserFolder
ControllerTagTable
ControllerTag
ControllerConstant
SystemblockSystemFolder
ProgramblockUserFolder
SystemblockComponentFolder
CodeBlock
DataBlock
ProjectLibrary
MasterCopySystemFolder
MasterCopyUserFolder
MasterCopy
HmiTarget
LibraryTypeSystemFolder
LibraryTypeUserFolder
ILibraryType
ILibraryTypeVersion
Connection
Cycle
GraphicList
TextList
VBScriptSystemFolder
VBScriptUserFolder
VBScript
TagSystemFolder
TagUserFolder
TagTable
Tag
ScreenSystemFolder
ScreenUserFolder
Screen
ScreenTemplateSystemFolder
ScreenTemplateUserFolder
ScreenTemplate
ScreenGlobalElements
ScreenOverview
GlobalLibrary

9.3.2 Vor dem Hochrüsten einer Anwendung auf TIA Portal Openness V14

Anwendung

Vor dem Hochrüsten einer Anwendung auf TIA Portal Openness V14 müssen die folgenden Einstellungen geändert werden:

1. Die Verweise auf die V14 API sind durch Ergänzen der folgenden TIA Portal Openness APIs anzupassen:
   - `Siemens.Engineering`
   - `Siemens.Engfineering.Hmi`
2. Das .Net-Framework von Visual Studio auf Version 4.6.1 ändern
3. Aktualisieren Sie die AssemblyResolve-Methode durch Anpassen des neuen Installationspfads des TIA Portal.
   - Arbeiten Sie aus der Registrierungsdatei, passen Sie den neuen Schlüssel wie im folgenden Beispiel an:
     `"HKEY_LOCAL_MACHINE\SOFTWARE\Siemens\Automation\_InstalledSW\TIAP14\TIA_Opns\..."`
   - Arbeiten Sie mit der Anwendungs-Konfigurationsdatei, passen Sie die Pfade an den neuen Installationspfad an.

9.3.3 Wesentliche Stringänderungen

Einleitung

An TIA Portal Openness V14 wurden die folgenden Änderungen vorgenommen, was möglicherweise Auswirkungen auf Ihre vorhandenen Anwendungen hat:

Änderung	Erforderliche Anpassung von Programmcode

Die Übersetzungsmethoden wurden geändert.	Ändern Sie die Übersetzungsmethoden wie im folgenden Beispiel:
	• TIA Portal Openness V13 SP1(veraltet):
	`controllerTarget.Compile(CompilerOptions.Software, BuildOptions.Rebuild);`
	• TIA Portal Openness V14:
	`plcSoftware.GetService<ICompilable>().Compile();`

Es wurden neue Namensräume hinzugefügt.	1. Fügen Sie die folgenden Namensraumanweisungen hinzu:
	`Siemens.Engineering.SW.Blocks;`
	`Siemens.Engineering.SW.ExternalSources;`
	`Siemens.Engineering.SW.Tags;`
	`Siemens.Engineering.SW.Types;`
	2. Entfernen Sie die Namensraumanweisung `using ControllerTarget = Siemens.Engineering.HW.ControllerTarget.`
	3. Übersetzen Sie die Anwendung.

`ControllerTarget` wurde durch `PlcSoftware` ersetzt und in einigen Fällen hat sich Funktionalität geändert.	1. Prüfen Sie die Codebeispiele in der Dokumentation, die zu Ihrer Anwendungsfunktionalität gehört.
	2. Aktualisieren Sie den Programmcode Ihrer TIA Portal Openness-Anwendung nach dem folgenden Beispiel:
	– TIA Portal Openness V13 SP1(veraltet):
	`ControllerTarget controllerTarget = deviceItem as ControllerTarget`
	– TIA Portal Openness V14:
	`PlcSoftware plcSoftware = deviceItem.GetService<SoftwareContainer>().Software as PlcSoftware`
	3. Übersetzen Sie die Anwendung.

Änderung	Erforderliche Anpassung von Programmcode

Es wurden Objekte ersetzt.	1. Suchen und ersetzen Sie die folgenden Objekte:
	`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`
	`IOnline = OnlineProvider`
	`ILibraryType = LibraryType`
	2. Übersetzen Sie die Anwendung.

Aggregationen wurden durch Zusammensetzungen ersetzt.	1. Ersetzen Sie jede `Aggregation` in Ihrem Code durch `Composition` wie in den folgenden Beispielen:
	`ProjectAggregation = ProjectComposition`
	`IDeviceAggregation = IDeviceComposition`
	`TagTableAggregation = TagTableComposition`
	`CycleAggregation = CycleComposition`
	`GraphicListAggregation = GraphicListComposition`
	`TextListAggregation = TextListComposition`
	`ConnectionAggregation = ConnectionComposition`
	`MultiLingualGraphicAggregation = MultiLingualGraphicComposition`
	`UpdateCheckResultMessageAggregation = UpdateCheckResultMessageComposition`
	2. Übersetzen Sie die Anwendung.

Ordner wurden in jeder Beziehung außer zu Bediengeräten durch Gruppen ersetzt.	1. Ersetzen Sie mit Ausnahme von Codeteilen, die Bediengeräte betreffen, jeden `Folder` in Ihrem Programmcode durch `Group`. 2. Übersetzen Sie die Anwendung.

Änderung	Erforderliche Anpassung von Programmcode
Die Methode `GetAttributeNames` wurde durch die Methode `GetAttributeInfos` ersetzt.	1. Verwenden Sie `IList<EngineeringAttributeInfo> IEngineeringObject.GetAttributeInfos(AttributeAccessMode attributeAccessMode);` , um Attribute zu ermitteln. 2. Übersetzen Sie die Anwendung. Ausführlichere Informationen finden Sie unter Objektstruktur und -Attribute ermitteln (Seite 111).
Die Methode `Close` zum Schließen eines Objekts hat sich geändert.	1. Ersetzen Sie `project.Close(CloseMode.PromptIfModified);` durch `project.Close();`. 2. Übersetzen Sie die Anwendung. Ausführlichere Informationen finden Sie unter Projekt schließen (Seite 122).

Gleichzeitiger Zugriff wurde durch exklusiven Zugriff und exklusive Transaktionen ersetzt.	1. Ersetzen Sie den gleichzeitigen Zugriff durch exklusiven Zugriff und exklusive Transaktionen wie in den folgenden Beispielen:
	– TIA Portal Openness V13 SP1(veraltet):
	`tiaProject.StartTransaction("Reseting project to default");`
	`...`
	`tiaProject.CommitTransaction();`
	– TIA Portal Openness V14:
	`//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();`
	2. Übersetzen Sie die Anwendung.
	Weitere Informationen hierzu siehe Exclusive access (Seite 90) und Transaktionsbehandlung (Seite 92).

Änderung	Erforderliche Anpassung von Programmcode

Der Online-Zugriff auf die CPU wurde geändert	1. Ändern Sie den Online-Zugriff auf die CPU wie in den folgenden Beispielen:
	– TIA Portal Openness V13 SP1(veraltet):
	`((IOnline)controllerTarget).GoOffline();`
	– TIA Portal Openness V14:
	`((DeviceItem) plcSoftware.Parent.Parent).GetService<Onlin eProvider>().GoOffline();`
	2. Übersetzen Sie die Anwendung.

Die Hardware-Konfiguration wurde geändert	1. Ändern Sie die Hardware-Konfiguration: `Device.Elements = Device.Items` 2. Entfernen Sie die folgenden Hardware-Attribute: – `Device.InternalDeviceItem` – `Device.SubType` 3. Übersetzen Sie die Anwendung.

Siehe auch

Umgang mit Exceptions (Seite 420)
Was ist neu in TIA Portal Openness? (Seite 23)
Verbindung zum TIA Portal aufbauen (Seite 74)

9.3.4 Importieren von Dateien, die mit TIA Portal Openness V13 SP1 und älter erzeugt wurden

Anwendung

Wenn Sie versuchen, Dateien zu importieren, die mit TIA Portal Openness V13 SP1 oder älter erzeugt wurden, wird eine Ausnahme wegen Inkompatibilität ausgelöst. Grund dafür sind Änderungen bei den HMI-Variablen und HMI-Bildschirmen. Die folgenden Tabellen zeigen die wichtigsten Attributänderungen. Ausführlichere Informationen finden Sie im Kapitel "Bilder erstellen Arbeiten mit Objekten und Objektgruppen > Arbeiten mit Objekten > Konfigurieren von Bereichen" in der Online-Hilfe des TIA Portal:

Änderungen von HMI-Variablen

Die folgende Tabelle zeigt die wichtigsten Änderungen der HMI-Variablenattribute:

Entfernte Attribute	Hinzugefügte Attribute
RangeMaximumType	LimitUpper2Type.
RangeMaximum	LimitUpper2.
RangeMinimumType	LimitLower2Type.
RangeMinimum	LimitLower2.
	LimitUpper1Type
	LimitUpper1
	LimitLower1Type
	LimitLower1

Änderungen von HMI-Bildelementen

Die folgende Tabelle zeigt die wichtigsten Änderungen der Schieberegler-Attribute:

Entfernte Attribute	Hinzugefügte Attribute
	RangeLower1Color
	RangeLower1Enabled
	RangeLower2Color
	RangeLower2Enabled
	RangeNormalColor
	RangeNormalEnabled
	RangeUpper1Color
	RangeUpper1Enabled
	RangeUpper2Color
	RangeUpper2Enabled
	ScalePosition
	ShowLimitLines
	ShowLimitMarkers
	ShowLimitRanges

Die folgende Tabelle zeigt die wichtigsten Änderungen der Messbereichsattribute:

Entfernte Attribute	Hinzugefügte Attribute
DangerRangeColor	RangeLower1Color
DangerRangeStart	RangeLower1Enabled
DangerRangeVisible	RangeLower2Color
WarningRangeColor	RangeLower2Enabled
WarningRangeStart	RangeNormalColor
WarningRangeVisible	RangeNormalEnabled
	RangeUpper1Color
	RangeUpper1Enabled
	RangeUpper1Start
	RangeUpper2Color
	RangeUpper2Enabled
	RangeUpper2Start

Die folgende Tabelle zeigt die wichtigsten Änderungen der Balkenattribute:

Entfernte Attribute	Hinzugefügte Attribute
AlarmLowerLimitColor	RangeLower1Color
AlarmUpperLimitColor	RangeLower1Enabled
	RangeLower2Color
	RangeLower2Enabled
	RangeNormalColor
	RangeNormalEnabled
	RangeUpper1Color
	RangeUpper1Enabled
	RangeUpper2Color
	RangeUpper2Enabled

Index

A

Abfragen
- Attribut „Consistency“ eines Bausteins, 311
- Bausteinautor, 311
- Bausteinfamilie, 311
- Bausteinname, 311
- Bausteinnummer, 311
- Bausteintitel, 311
- Bausteintyp, 311
- Bausteinversion, 311
- Finden, 305
- Informationen aus einer PLC-Variablentabelle, 371
- Informationen des Anwenderdatentyps, 311
- Informationen des Bausteins, 311
- Ordner „Programmbausteine“, 305
- Systemordner für PLC-Variablen, 367
- Technologieobjekt, 329
- Zeitstempel eines Bausteins, 311

Allgemeine Einstellungen des TIA Portals, 104
Anwenderdatentyp
- Exportieren, 501
- Importieren, 559
- Informationen abfragen, 311
- Löschen, 324
- Quelle generieren, 320

Anwendungsbeispiel Public API, 67
Ausnahmen
- Beim Zugriff auf das TIA Portal über öffentliche APIs, 420

B

Baustein
- Exportieren, 501
- Gruppe erzeugen, 315
- Gruppe löschen, 315
- Importieren, 550
- Informationen abfragen, 311
- Löschen, 314
- Quelle generieren, 320

Bausteineditor
- Starten, 325

Bearbeitungssituation
- Ihre Openness-Anwendung und das TIA Portal befinden sich auf demselben Computer, 46

Beenden der Verbindung zum TIA Portal, 83
Beispielprogramm, 49
Besondere Überlegungen für HMI-Variablen des Datentyps "UDT", 452
Bibliothek
- Auf Ordner zugreifen, 139
- Funktionen, 127
- Typversionen von Instanzen ermitteln, 159

D

Datentypen
- Technologieobjekt, 328

E

Editor „Geräte & Netze“
- Öffnen, 168

Enumerieren
- Alle Variablen einer Variablentabelle, 259
- Bausteine, 309
- Benutzerdefinierte Bausteinordner, 308
- Benutzerdefinierte Ordner für PLC-Variablen, 368
- Geräte, 220, 223
- Geräteelemente, 236
- Mehrsprachige Texte, 109, 114
- Parameter eines Technologieobjekts, 334
- PLC-Variablen, 374
- PLC-Variablentabellen, 370
- Systemunterordner, 306
- Technologieobjekt, 333

Erstellen
- Benutzerdefinierte Bilderordner erzeugen, 253
- Benutzerdefinierte Ordner für HMI-Variablen, 258
- Benutzerdefinierte Unterordner für Skripte, 261
- Benutzerdefinierter Ordner für PLC-Variablentabellen, 369
- Gruppe für Baustein, 315
- Messtaster, 345
- Nocken, 345
- Nockenspur, 345
- Technologieobjekt, 330

Export/Import
- Verwendung, 36

Exportdatei
- Grundstruktur, 439, 564, 568
- Inhalt, 428
- Struktur der XML-Datei, 439, 568

Exportierbare Bildobjekte, 462
Exportieren
Anwenderdatentyp, 501
Baustein, 501
Einzelne Variable oder Konstante aus einer PLC-Variablentabelle, 556

F

Finden
Messtaster, 345
Nocken, 345
Nockenspur, 345
Parameter eines Technologieobjekts, 335
Technologieobjekt, 333
Funktionen, 49
Alle Bilder löschen, 255
Allgemein, 74, 82, 83
Allgemeine Einstellungen des TIA Portals, 104
Anwendungsbeispiel Public API, 67
Attribut „Consistency“ eines Bausteins abfragen, 311
Bausteinautor abfragen, 311
Bausteine enumerieren, 309
Bausteinfamilie abfragen, 311
Bausteinname abfragen, 311
Bausteinnummer abfragen, 311
Bausteintitel abfragen, 311
Bausteintyp abfragen, 311
Bausteinversion abfragen, 311
Begrenzung auf Projekte von TIA Portal V13, 99
Benutzerdefinierte Bausteinordner enumerieren, 308
Benutzerdefinierte Bilderordner erzeugen, 253
Benutzerdefinierte Ordner für HMI-Variablen erzeugen, 258
Benutzerdefinierte Ordner für PLC-Variablen enumerieren, 368
Benutzerdefinierte Unterordner für Skripte erzeugen, 261
Benutzerdefinierten Ordner für PLC-Variablentabellen erzeugen, 369
Benutzerdefinierten Ordner für PLC-Variablentabellen löschen, 370
Bild löschen, 253
Bildvorlage löschen, 254
Einlesen der Uhrzeit der letzten Änderungen in eine PLC-Variablentabelle, 373
Geräte enumerieren, 220, 223
Geräteelemente enumerieren, 236
Grafikliste löschen, 257
Grafiksammlungen löschen, 118
HMI, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262
Informationen aus einer PLC-Variablentabelle abfragen, 371
Mehrsprachige Texte enumerieren, 109, 114
Ordner „Programmbausteine“ abfragen, 305
PLC, 305, 306, 308, 309, 311, 369, 370, 373, 375, 376, 555, 556, 557
PLC-Konstanten, 376
PLC-Target und HMI-Target abfragen, 169
PLC-Variablen enumerieren, 374
PLC-Variablentabelle importieren, 555
PLC-Variablentabelle löschen, 373
PLC-Variablentabellen in Ordnern enumerieren, 370
Projekt öffnen, 99
Projekt schließen, 122
Projekt speichern, 121
Projekte, 99, 104, 109, 114, 118, 121, 122, 169, 220, 223, 236, 367, 368, 370, 371, 374
Systemordner ermitteln, 305
Systemordner für PLC-Variablen abfragen, 367
Systemunterordner enumerieren, 306
Textliste löschen, 256
Variable aus einer PLC-Variablentabelle löschen, 375
Variable aus einer Variablentabelle löschen, 260
Variable in eine PLC-Variablentabelle importieren, 557
Variable oder Konstante aus einer PLC-Variablentabelle exportieren, 556
Variablen einer HMI-Variablentabelle enumerieren, 259
Variablentabelle löschen, 260
VB-Skript aus einem Ordner löschen, 262
Verbindung löschen, 258
Zeitstempel eines Bausteins abfragen, 311
Zyklus löschen, 256

G

Generieren
Quelle aus Anwenderdatentyp, 320
Quelle aus Baustein, 320
Geräte enumerieren, 220, 223
Geräteelemente enumerieren, 236
Globale Bibliothek
Zugreifen, 128, 132
Zugriff auf Spracheinstellungen, 130
Grundstruktur einer AML-Exportdatei, 564
Grundstruktur einer Exportdatei, 439, 568

H

Hardware
Übersetzen, 119
Hierarchie von Hardware-Objekten des Objektmodells, 64
HMI-Variablen des Datentyps "UDT", 452

I

Import/Export
XML-Datei bearbeiten, 427
Importieren
Anwenderdatentyp, 559
Baustein, 550
Einzelne Variable in eine PLC-Variablentabelle, 557
PLC-Variablentabellen, 555
Importieren/Exportieren
Alle Bildvorlagen exportieren, 473
Alle Grafiken eines Projekts exportieren, 432
Anwendungsbereich, 425
Auch Standardwerte exportieren, 428
Ausgewählte Variable exportieren, 449
Bausteine mit Knowhow-Schutz exportieren, 509
Bausteine ohne Knowhow-Schutz exportieren, 501
Besondere Überlegungen für integrierte HMI-Variablen, 451
Bild aus einem Bilderordner exportieren, 467
Bild mit einer Bildbausteininstanz exportieren, 483
Bild mit einer Bildbausteininstanz importieren, 485
Bilder eines HMI-Geräts exportieren, 466
Bilder in ein HMI-Gerät importieren, 469
Bildvorlagen exportieren, 474
Bildvorlagen importieren, 476
Datenstruktur, 439, 568
Eine HMI-Variable in eine Variablentabelle importieren, 450
Einschränkungen, 425
Erweiterte XML-Formate für den Export/Import von Textlisten, 457
Exporte auf geänderte Werte beschränken, 428
Exporteinstellungen, 427
Exportformat, 425
Exportierbare Bildobjekte, 462
Exportierbare Objekte, 423
Exportieren eines Slide-in-Bilds, 480
Exportieren mehrsprachiger Kommentare, 607, 612, 621, 623
Exportieren von Pop-up-Bildern, 478
Exportumfang, 427
Grafiken, 431
Grafiken in ein Projekt importieren, 433
Grafikliste importieren, 459
Grafiklisten exportieren, 459
Grundlagen, 423
HMI, 443, 444, 445, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 459, 461, 462, 466, 467, 469, 472, 473, 474, 476, 478, 479, 480, 482, 483, 485, 554
HMI-Variablentabellen exportieren, 445
Importierbare Objekte, 423
Importieren eines Slide-in-Bilds, 482
Importieren mehrsprachiger Kommentare, 607, 612, 621, 623
Importieren von Pop-up-Bildern, 479
Importverhalten mit Programmcodes einstellen, 430
Konfigurationsdaten exportieren, 427
Konfigurationsdaten importieren, 429
Nur geänderte Werte exportieren, 428
Objekte von AML, 564
Permanentfenster exportieren, 472
Permanentfenster importieren, 472
PLC, 501, 509, 547
PLC-Variablentabelle exportieren, 554
Projektdaten, 432, 433
Round-Trip-Geräte und -Module, 585
Stabile AML-GUIDs, 585
Systembausteine exportieren, 547
Textlisten exportieren, 455
Textlisten importieren, 456
Variable aus einer Variablentabelle exportieren, 449
Variablen exportieren, 619
Variablen importieren, 619
Variablentabelle in einen Variablenordner importieren, 448
VB-Skripte exportieren, 452, 453
VB-Skripte importieren, 454
Verbindungen exportieren, 460
Verbindungen importieren, 461
Vorgehensweise zum Importieren, 430
Zyklen exportieren, 443
Zyklen importieren, 444
Installation
Authentifizierung für Zugriff prüfen, 28
Benutzer der Benutzergruppe hinzufügen, 28

Standardschritte für den Zugriff auf das TIA Portal, 34
TIA Openness V13 Add-on-Paket, 27
Installation des Add-on-Pakets, 27
Instanzen
Typversion ermitteln, 159
Integrierte HMI-Variablen, 451

K

Konfiguration
Ihre Openness-Anwendung und das TIA Portal befinden sich auf unterschiedlichen Computern, 45
Kopieren
Inhalt einer Masterkopie in Projektordner, 156
Masterkopie, 159

L

Lesen
Parameter eines Technologieobjekts, 336
Uhrzeit der letzten Änderungen an einer PLC-Variablentabelle, 373
Löschen
Alle Bilder, 255
Anwenderdatentyp, 324
Baustein, 314
Benutzerdefinierter Ordner für PLC-Variablentabellen, 370
Bild, 253
Bildvorlage, 254
Einzelne Variable in einer PLC-Variablentabelle, 375
Einzelne Variablen einer Variablentabelle, 260
Grafikliste, 257
Grafiksammlung, 118
Gruppe für Baustein, 315
PLC-Konstanten, 376
PLC-Variablentabelle aus einem Ordner löschen, 373
Programmbaustein, 314
Technologieobjekt, 331
Textliste, 256
Variablentabelle, 260
VB-Skript aus einem Ordner, 262
Verbindung, 258
Zyklus, 256

M

Masterkopie
Inhalt in Projektordner kopieren, 156
Kopieren, 159
Masterkopien
Löschen, 165
Mehrsprachige Texte enumerieren, 109, 114

O

Objekte
Exportierbare Objekte, 423
Importierbare Objekte, 423
Objektmodell, 51
Öffnen
Editor „Geräte & Netze“, 168
Öffnen eines Projekts, 99
Ordner
Löschen, 165

P

Parameter
Easy Motion Control, 365
PID-Regelung, 364
S7-1500 Motion Control, 347
Zählung, 365
Parameter eines Technologieobjekts
Enumerieren, 334
Finden, 335
Lesen, 336
Schreiben, 337
PLC
Online-Verbindung herstellen, 303
Online-Verbindung trennen, 303
Status ermitteln, 263
Vergleich mit tatsächlichem Status, 299
Vergleichen, 299
Programmbaustein
Löschen, 314
Programmgesteuertes Quittieren von Systemereignissen, 82
Programmierübersicht, 49
Projekt
Gerätetyp abfragen, 169
HMI-Targets abfragen, 169
Öffnen, 99
PLC-Targets abfragen, 169

Schließen, 122
Speichern, 121
Projekt speichern, 121
Projektbibliothek
Auf Masterkopien zugreifen, 152
Zugreifen, 128, 132

S

Schreiben
Parameter eines Technologieobjekts, 337
Siemens.Engineering, 41
Siemens.Engineering.Hmi, 41
Siemens.Engineering.Hmi.Communication, 41
Siemens.Engineering.Hmi.Cycle, 41
Siemens.Engineering.Hmi.Globalization, 41
Siemens.Engineering.Hmi.RuntimeScripting, 41
Siemens.Engineering.Hmi.Screen, 41
Siemens.Engineering.Hmi.Tag, 41
Siemens.Engineering.Hmi.TextGraphicList, 41
Siemens.Engineering.HW, 41
Siemens.Engineering.SW, 41
Software
Übersetzen, 119
Starten
Bausteineditor, 325
Variableneditor, 366
Status (PLC)
Ermitteln, 263
Struktur der Exportdaten, 439, 564, 568

T

Technologieobjekt, 326
Abfragen, 329
Datentypen, 328
Enumerieren, 333
Erstellen, 330
Finden, 333
Löschen, 331
Übersetzen, 331
Technologieobjektgruppe
Übersetzen, 332
TIA Portal Openness, 43
Benutzer der Benutzergruppe hinzufügen, 28
Einleitung, 43
Export/Import, 36
Funktionen, 49
Funktionsumfang, 43
Grundbegriffe von Aggregationen, 175
Grundlegende Konzepte bei der Handhabung von Ausnahmen, 420
Grundlegende Konzepte der Überprüfung auf Objektgleichheit, 177
Grundlegende Konzepte von Zuordnungen, 175
Konfiguration, 45
Notwendige Kenntnisse des Benutzers, 25
Programmierübersicht, 49
Public API, 49
Standardschritte für den Zugriff auf das TIA Portal, 34
Typische Aufgaben, 35
Voraussetzungen, 25
Zugriff, 35
Zugriffsrechte, 28
Typen
Löschen, 165

U

Übersetzen
Hardware, 119
Software, 119
Technologieobjekt, 331
Technologieobjektgruppe, 332

V

Variableneditor
Starten, 366
Verbindung zum TIA Portal
Einrichtung, 74
Schließen, 83
Verbindung zum TIA Portal herstellen, 74
Verschalten
Analogantriebe mit der Hardware-Adresse, 341
Analogantriebe mit einem Datenbaustein, 343
Antriebe, 351
Geber, 356
Geber für Analogantriebe mit der Hardware-Adresse, 342
Geber für PROFIdrives mit der Hardware-Adresse, 340
Geber mit einem Datenbaustein, 344
Gleichlaufachse mit Leitwerten, 361
Messtaster, 359
Nocken, 358
Nockenspur, 358
PROFIdrives mit der Hardware-Adresse, 339
PROFIdrives mit einem Datenbaustein, 343
Telegramm 750, 354

X

XML-Datei
Bearbeiten, 427
Export, 428

Z

Zugreifen
Masterkopie in einer Projektbibliothek, 152