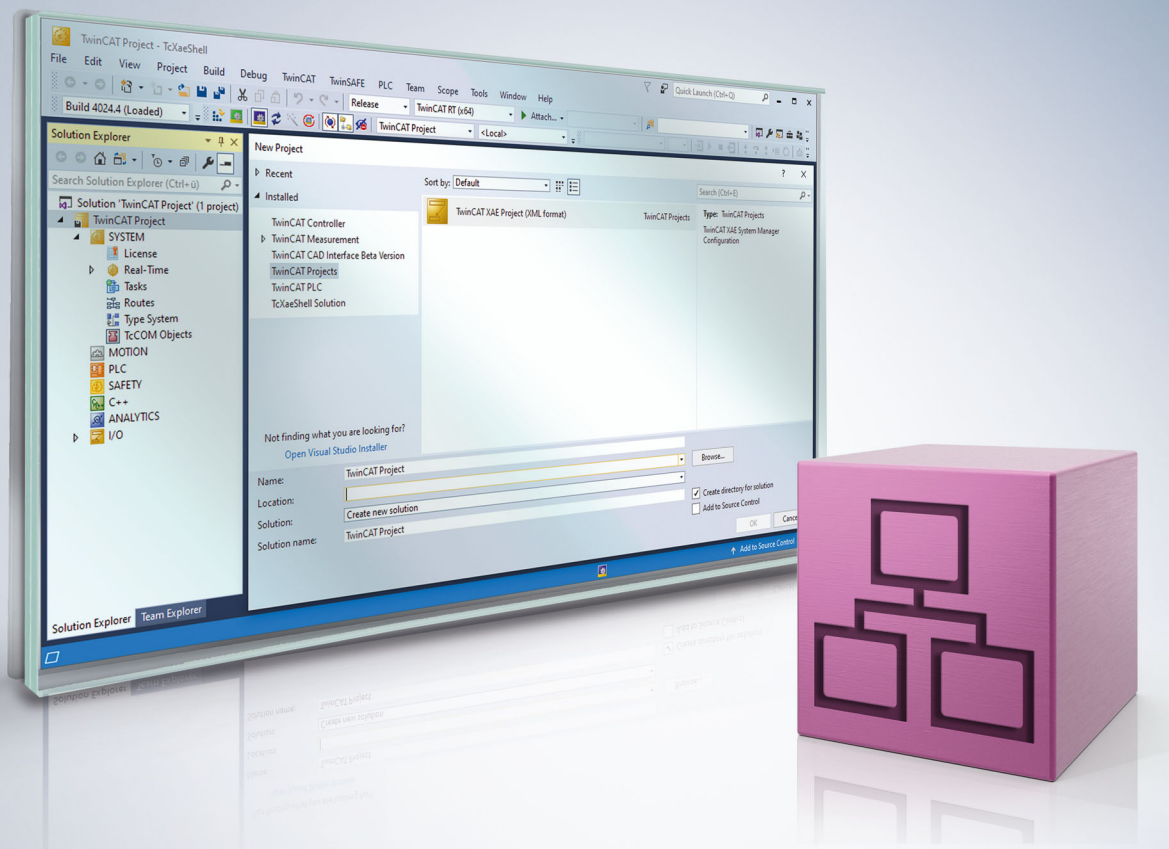
BECKHOFF New Automation Technology

Handbuch | DE

TF6421

TwinCAT 3 | XML Server



28.11.2022 | Version: 1.4.1

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT®

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

Beachten Sie die folgenden Sicherheitshinweise und Erklärungen!
Produktspezifische Sicherheitshinweise finden Sie auf den folgenden Seiten oder in den Bereichen Montage, Verdrahtung, Inbetriebnahme usw.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Erklärung der Symbole

In der vorliegenden Dokumentation werden die folgenden Symbole mit einem nebenstehenden Sicherheitshinweis oder Hinweistext verwendet. Die Sicherheitshinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ WARNUNG

Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, können Personen geschädigt werden!

HINWEIS

Schädigung von Umwelt oder Geräten

Wenn der Hinweis neben diesem Symbol nicht beachtet wird, können Umwelt oder Geräte geschädigt werden.

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem https://www.beckhoff.de/secguide.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter https://www.beckhoff.de/secinfo.

2 Übersicht

Der TwinCAT XML Server bietet eine SPS-Bibliothek, mit der ein Schreib- und Lesezugriff auf XML-Dateien realisiert werden kann. Der XML Server zeichnet sich durch seine einfache Handhabung aus. Besonders eignet er sich beispielsweise für das Laden von Initialisierungsdaten, so wie es häufig beim Aufstarten einer Maschine benötigt wird. Außerdem ist es möglich, SPS-Variablen formatiert in einer XML-Datei inklusive SPS-Kommentar zu speichern. Die Struktur einer Variablen im XML-Dokument entspricht dabei der Struktur der Variablen in der SPS. So ist es möglich, auch direkt auf einzelne Unterelemente einer Variablen zuzugreifen. Dabei werden nur die Unterelemente (Elemente einer Struktur oder Elemente eines Arrays) übertragen, die auch in der XML-Datei definiert sind. Beim Schreiben der SPS-Variablen gibt es jedoch auch die Möglichkeit, fehlende Elemente hinzuzufügen.

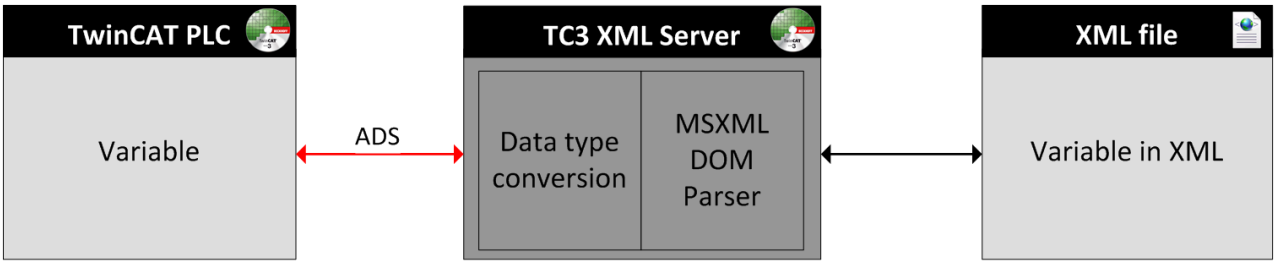
Komponenten

- TwinCAT XML Server: Ist ein Service der mit TwinCAT zusammen gestartet und gestoppt wird. Er ist das Bindeglied zwischen TwinCAT und der XML-Datei.
- SPS-Bibliothek [▶ 17]: Die SPS-Bibliothek bietet vier verschiedene Funktionsbausteine, mit denen Daten aus der SPS-Bibliothek in die XML-Datei geschrieben und aus ihr ausgelesen werden können. So können Variablen formatiert in einer XML-Datei gespeichert werden und Variablen in TwinCAT aus einer XML-Datei initialisiert werden.

Funktionsprinzip

Der XML Server kommuniziert mit der TwinCAT PLC über ADS. Bei einem Schreibvorgang werden alle zu schreibenden Variablen in Text konvertiert und über den MSXML DOM Parser in die XML-Datei geschrieben. Die XML-Datei beinhaltet keine Informationen über die Datentypen, sondern lediglich den Namen der Variablen als Tag-Namen und den Wert: <Variablenname> Wert </Variablenname>.

Bei einem Lesevorgang werden für die zu lesenden Variablen die Datentypen über ADS ermittelt und die Texte aus der XML-Datei entsprechend konvertiert.



3 Installation

3.1 Systemvoraussetzungen

Technische Daten	TF6421 XML Server
Zielsystem	Windows XP, Windows 7/8, Windows CE PC (x86-kompatibel), ARM
Min. TwinCAT-Version	3.1 Build 4011
Min. TwinCAT-Level	TwinCAT PLC

3.2 Installation

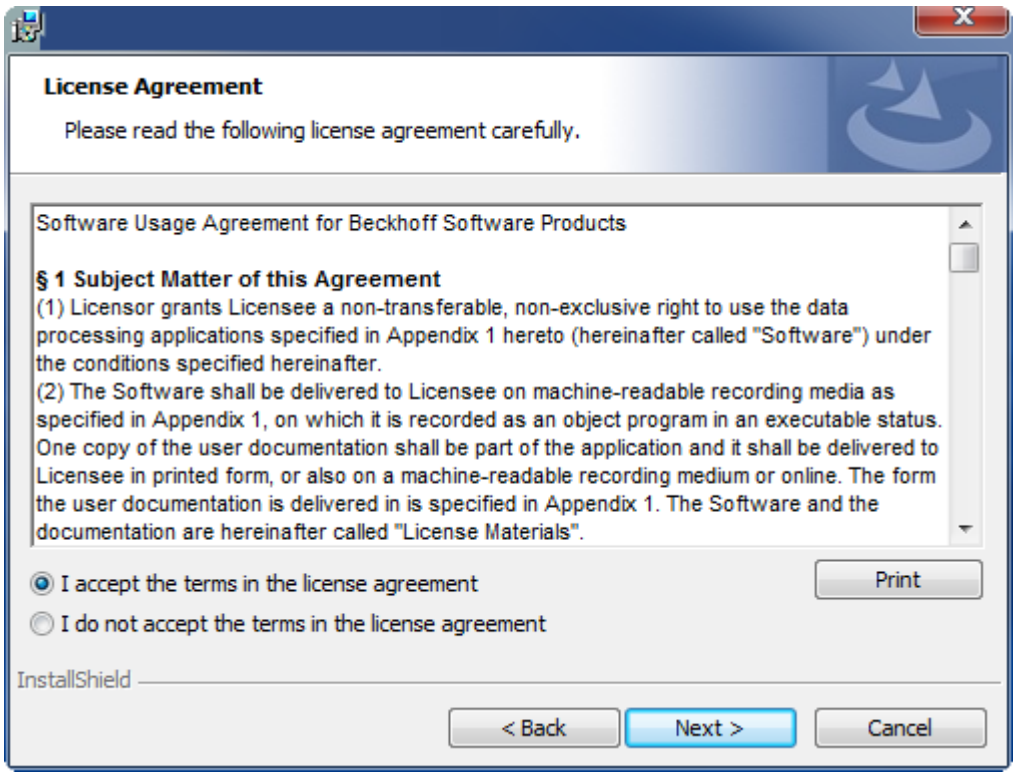
Nachfolgend wird beschrieben, wie die TwinCAT 3 Function für Windows-basierte Betriebssysteme installiert wird.

✓ Die Setup-Datei der TwinCAT 3 Function wurde von der Beckhoff-Homepage heruntergeladen.

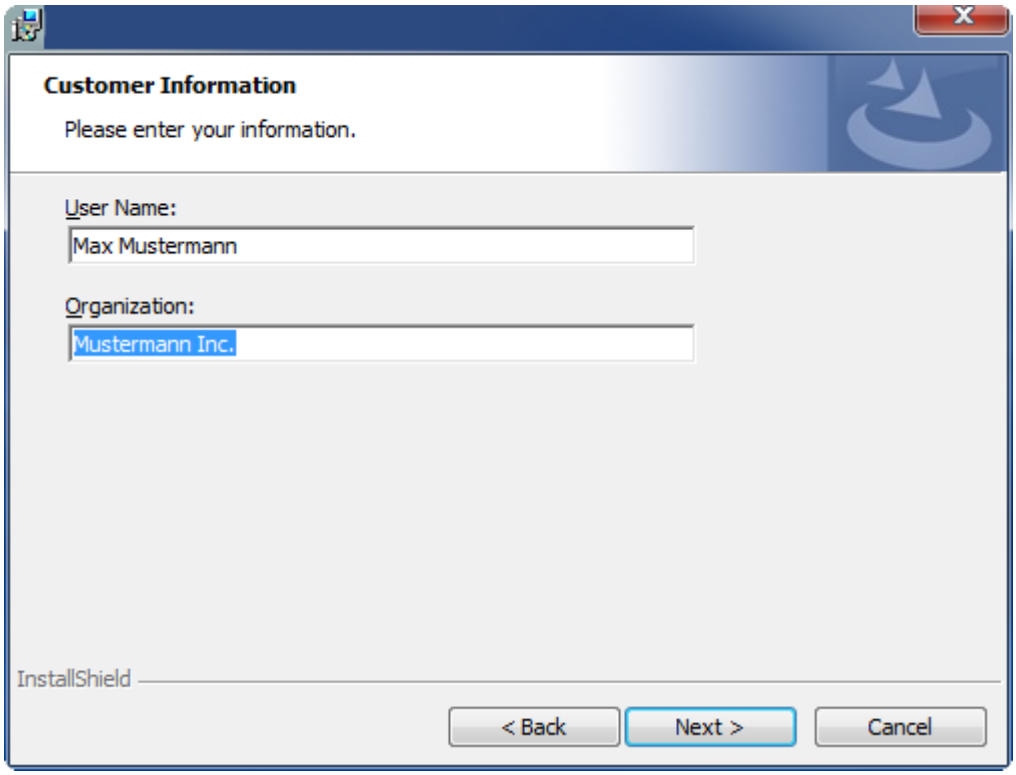
1. Führen Sie die Setup-Datei als Administrator aus. Wählen Sie dazu im Kontextmenü der Datei den Befehl **Als Administrator ausführen**.

   ⇨ Der Installationsdialog öffnet sich.

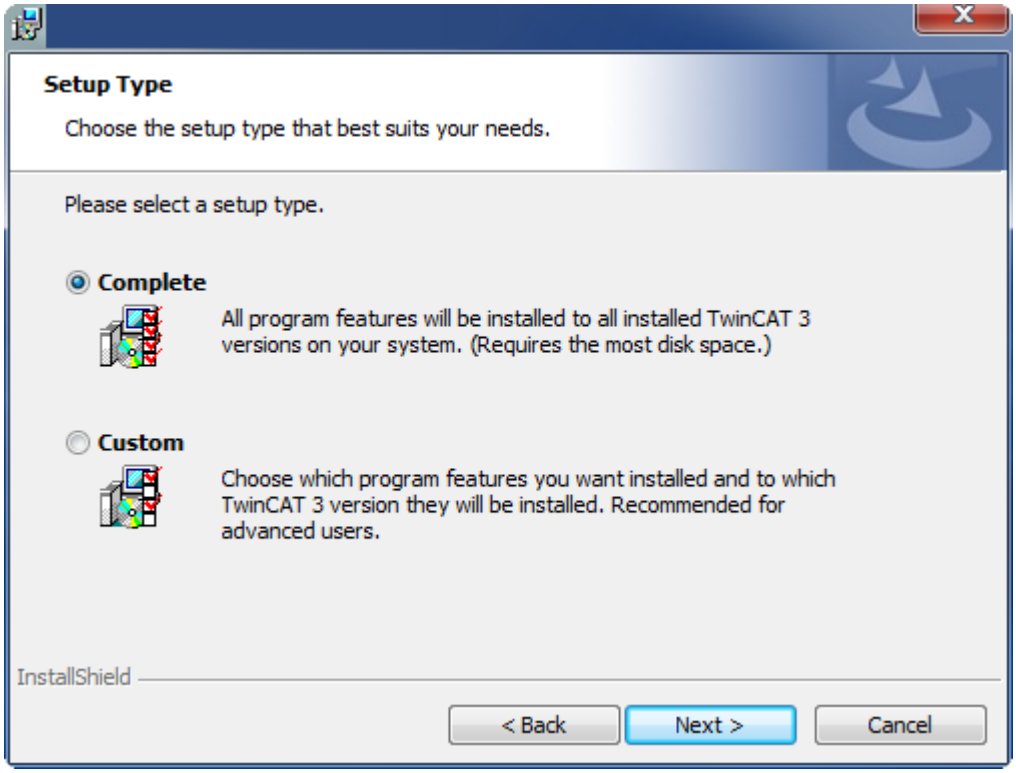
2. Akzeptieren Sie die Endbenutzerbedingungen und klicken Sie auf **Next**.



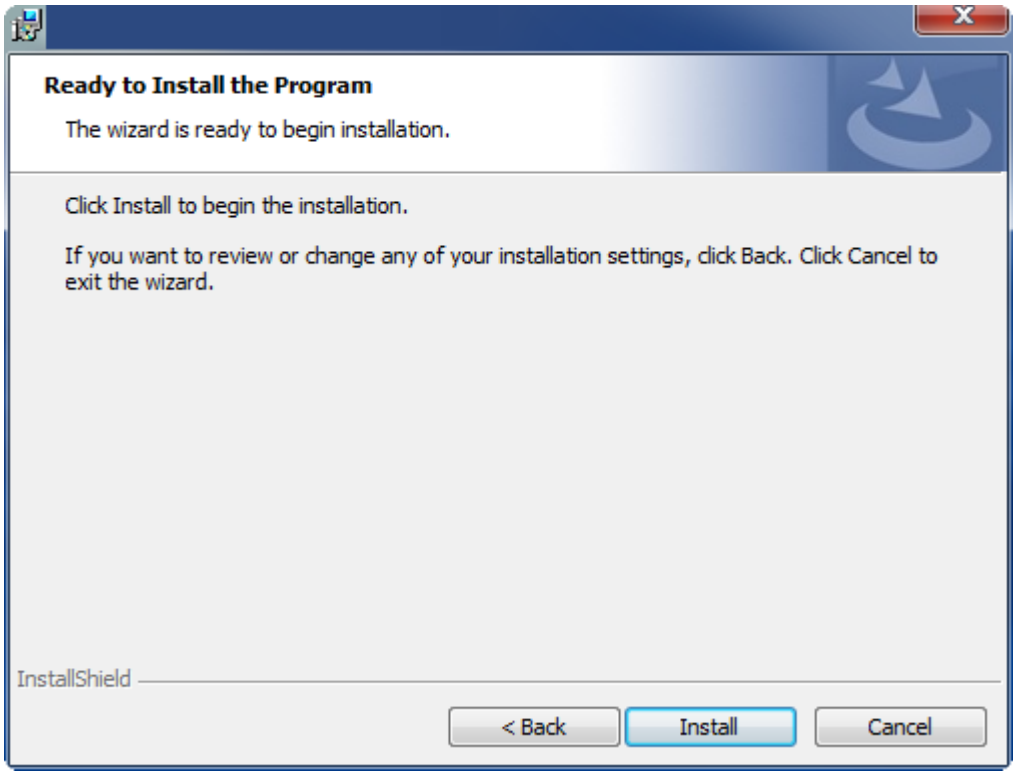
3. Geben Sie Ihre Benutzerdaten ein.



4. Wenn Sie die TwinCAT 3 Function vollständig installieren möchten, wählen Sie **Complete** als Installationstyp. Wenn Sie die Komponenten der TwinCAT 3 Function separat installieren möchten, wählen Sie **Custom**.

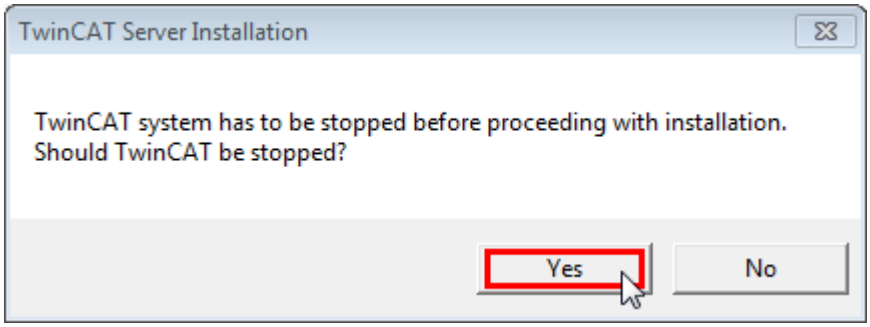


5. Wählen Sie **Next** und anschließend **Install**, um die Installation zu beginnen.

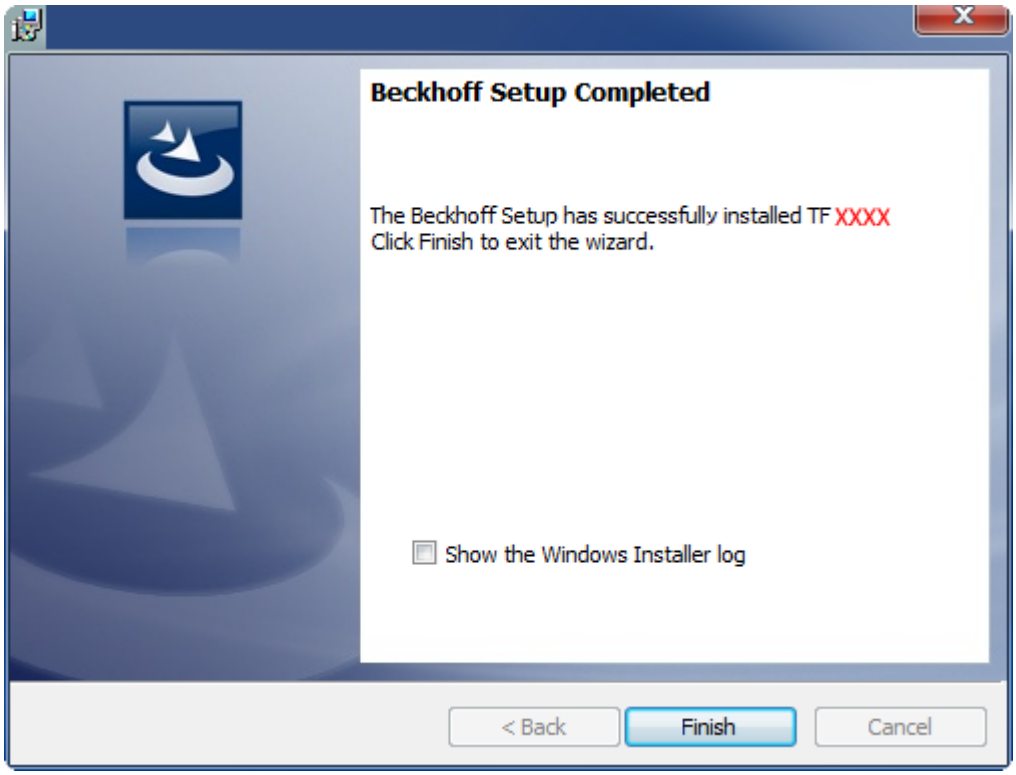


⇨ Ein Dialog weist Sie darauf hin, dass das TwinCAT-System für die weitere Installation gestoppt werden muss.

6. Bestätigen Sie den Dialog mit **Yes.**



7. Wählen Sie **Finish**, um das Setup zu beenden.



⇨ Die TwinCAT 3 Function wurde erfolgreich installiert und kann lizenziert werden (siehe Lizenzierung [▶ 12]).

3.3 Lizenzierung

Die TwinCAT 3 Function ist als Vollversion oder als 7-Tage-Testversion freischaltbar. Beide Lizenztypen sind über die TwinCAT-3-Entwicklungsumgebung (XAE) aktivierbar.

Lizenzierung der Vollversion einer TwinCAT 3 Function

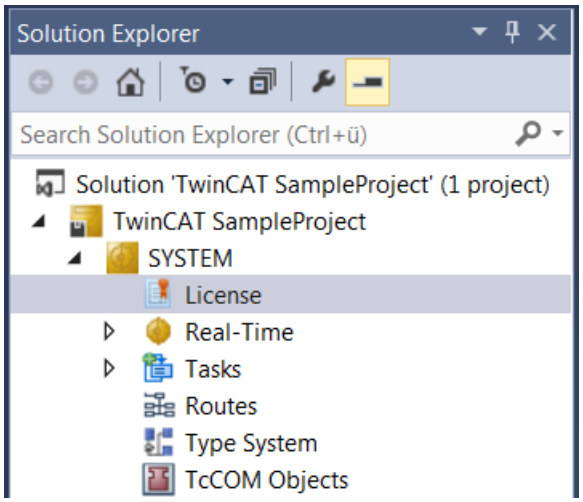
Die Beschreibung der Lizenzierung einer Vollversion finden Sie im Beckhoff Information System in der Dokumentation „TwinCAT 3 Lizenzierung“.

Lizenzierung der 7-Tage-Testversion einer TwinCAT 3 Function

Eine 7-Tage-Testversion kann nicht für einen TwinCAT 3 Lizenzdongle freigeschaltet werden.

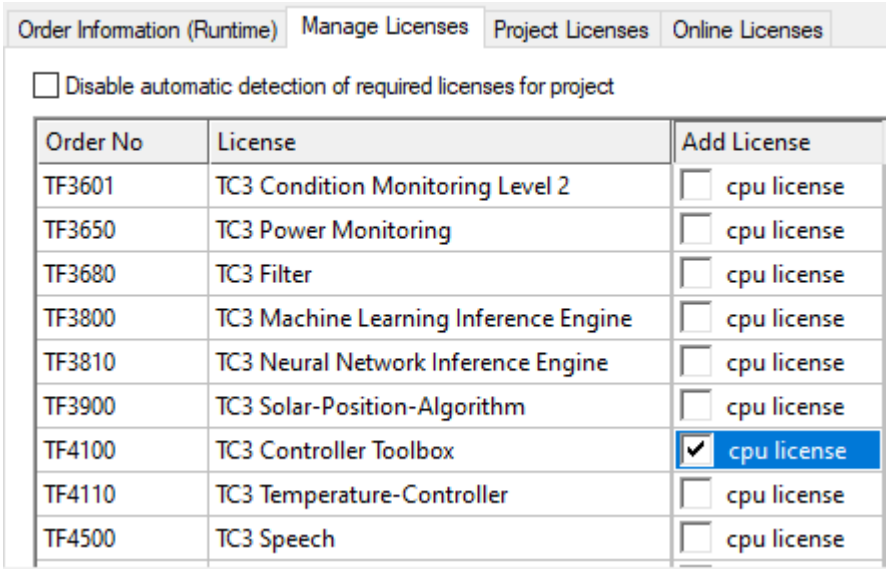
1. Starten Sie die TwinCAT-3-Entwicklungsumgebung (XAE).
2. Öffnen Sie ein bestehendes TwinCAT-3-Projekt oder legen Sie ein neues Projekt an.
3. Wenn Sie die Lizenz für ein Remote-Gerät aktivieren wollen, stellen Sie das gewünschte Zielsystem ein. Wählen Sie dazu in der Symbolleiste in der Drop-down-Liste **Choose Target System** das Zielsystem aus.
   ⇨ Die Lizenzierungseinstellungen beziehen sich immer auf das eingestellte Zielsystem. Mit der Aktivierung des Projekts auf dem Zielsystem werden automatisch auch die zugehörigen TwinCAT-3-Lizenzen auf dieses System kopiert.

4. Klicken Sie im **Solution Explorer** im Teilbaum **SYSTEM** doppelt auf **License.**



⇨ Der TwinCAT-3-Lizenzmanager öffnet sich.

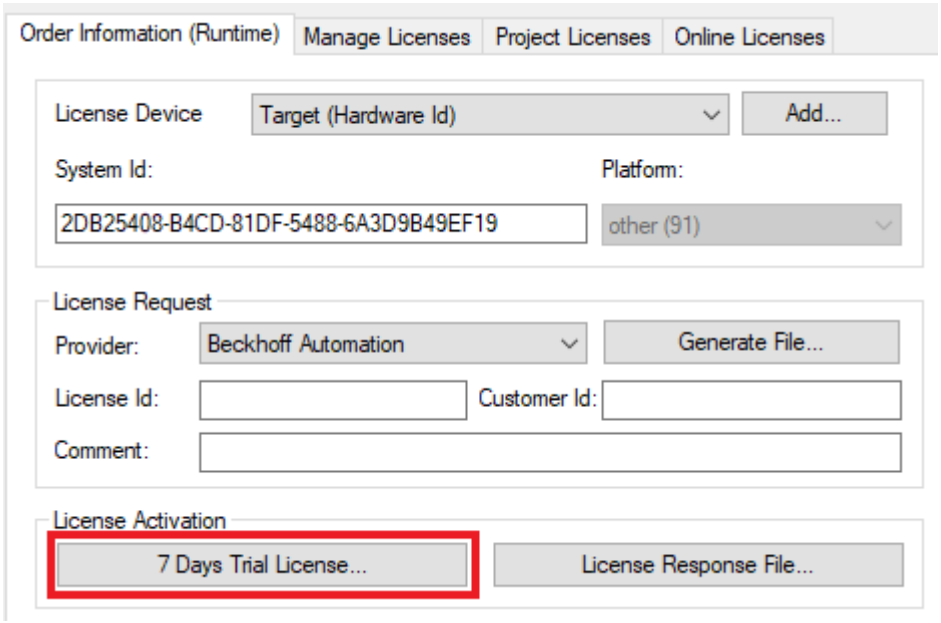
5. Öffnen Sie die Registerkarte **Manage Licenses.** Aktivieren Sie in der Spalte **Add License** das Auswahlkästchen für die Lizenz, die Sie Ihrem Projekt hinzufügen möchten (z. B. „TF4100 TC3 Controller Toolbox“).



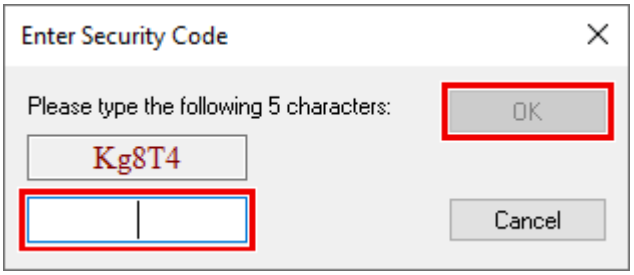
6. Öffnen Sie die Registerkarte **Order Information (Runtime).**

⇨ In der tabellarischen Übersicht der Lizenzen wird die zuvor ausgewählte Lizenz mit dem Status „missing“ angezeigt.

7. Klicken Sie auf **7 Days Trial License...**, um die 7-Tage-Testlizenz zu aktivieren.



⇨ Es öffnet sich ein Dialog, der Sie auffordert, den im Dialog angezeigten Sicherheitscode einzugeben.



8. Geben Sie den Code genauso ein, wie er angezeigt wird, und bestätigen Sie ihn.
9. Bestätigen Sie den nachfolgenden Dialog, der Sie auf die erfolgreiche Aktivierung hinweist.
   ⇨ In der tabellarischen Übersicht der Lizenzen gibt der Lizenzstatus nun das Ablaufdatum der Lizenz an.
10. Starten Sie das TwinCAT-System neu.

⇨ Die 7-Tage-Testversion ist freigeschaltet.

3.4 Installation Windows CE

Nachfolgend wird beschrieben, wie eine TwinCAT 3 Function (TFxxx) auf einem Beckhoff Embedded-PC mit Windows CE installiert wird.

1. Download der Setup-Datei und Installation [▶ 14]
2. CAB-Datei auf das Windows-CE-Gerät übertragen [▶ 15]
3. CAB-Datei auf dem Windows-CE-Gerät ausführen [▶ 15]

Wenn bereits eine ältere TFxxx-Version auf dem Windows-CE-Gerät installiert ist, kann diese aktualisiert werden:

- Upgrade der Software [▶ 15]

Download der Setup-Datei und Installation

Die CAB-Installationsdatei für Windows CE ist Teil des TFxxx-Setups. Dieses wird Ihnen auf der Beckhoff-Homepage www.beckhoff.com zur Verfügung gestellt und enthält automatisch alle Versionen für Windows XP, Windows 7 und Windows CE (x86 und ARM).

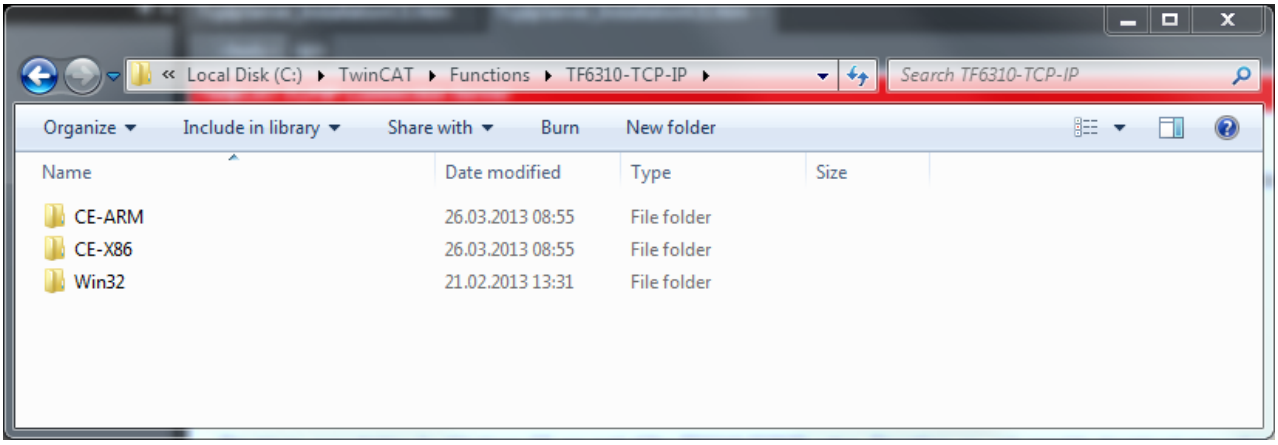
Laden Sie das TFxxx-Setup herunter und installieren Sie die TwinCAT 3 Function wie es im Abschnitt Installation [▶ 9] beschrieben wird.

Nach der Installation enthält der Installationsordner drei Verzeichnisse (pro Hardware-Plattform ein Verzeichnis):

- **CE-ARM:** ARM-basierte Embedded-PC, die unter Windows CE laufen, z. B. CX8090, CX9020
- **CE-X86:** X86-basierte Embedded-PC, die unter Windows CE laufen, z. B. CX50xx, CX20x0
- **Win32:** Embedded-PC, die unter Windows XP, Windows 7 oder Windows Embedded Standard laufen

Die Verzeichnisse CE-ARM und CE-X86 enthalten die CAB-Dateien der TwinCAT 3 Function für Windows CE in Bezug auf die jeweilige Hardware-Plattform des Windows-CE-Gerätes.

Beispiel: Installationsordner „TF6310“



CAB-Datei auf das Windows-CE-Gerät übertragen

Übertragen Sie die entsprechende CAB-Datei auf das Windows-CE-Gerät.

Für die Übertragung der ausführbaren Datei stehen Ihnen verschiedene Möglichkeiten zur Verfügung:

- über Netzwerkfreigaben
- über den integrierten FTP-Server
- über ActiveSync
- über CF/SD-Karten

Weitere Informationen finden Sie im Beckhoff Information System in der Dokumentation „Betriebssysteme“ (Embedded-PC > Betriebssysteme > CE).

CAB-Datei auf dem Windows-CE-Gerät ausführen

Nachdem Sie die CAB-Datei auf das Windows-CE-Gerät übertragen haben, führen Sie die Datei dort mit einem Doppelklick aus. Bestätigen Sie den Installationsdialog mit **OK**. Starten Sie das Windows-CE-Gerät anschließend neu.

Nach dem Neustart des Gerätes werden die Dateien der TwinCAT 3 Function (TFxxxx) automatisch im Hintergrund geladen und sind verfügbar.

Die Software wird in dem folgenden Verzeichnis auf dem Windows-CE-Gerät installiert:
\Hard Disk\TwinCAT\Functions\TFxxxx

Upgrade der Software

Wenn auf dem Windows-CE-Gerät bereits eine ältere Version der TwinCAT 3 Function installiert ist, führen Sie die folgenden Schritte auf dem Windows-CE-Gerät durch, um ein Upgrade auf eine neue Version durchzuführen:

1. Öffnen Sie den CE Explorer, indem Sie auf **Start > Run** klicken und „Explorer“ eingeben.

2. Navigieren Sie nach *\Hard Disk\TwinCAT\Functions\TFxxx\xxxx*.
3. Benennen Sie die Datei *Tc*.exe* in *Tc*.old* um.
4. Starten Sie das Windows-CE-Gerät neu.
5. Übertragen Sie die neue CAB-Datei auf das Windows-CE-Gerät.
6. Führen Sie die CAB-Datei auf dem Windows-CE-Gerät aus und installieren Sie die neue Version.
7. Löschen Sie die Datei *Tc*.old*.
8. Starten Sie das Windows-CE-Gerät neu.

⇨ Nach dem Neustart ist die neue Version aktiv.

4 SPS-Bibliotheken

4.1 Übersicht

Die SPS-Bibliothek **Tc2_XmlDataSrv** wird mit dem TC3 XML Server mitgeliefert und während der Installation in den Ordner *...C:\TwinCAT\3.1\Components\Plc\Managed Libraries\Beckhoff Automation GmbH* kopiert.

Es gibt jeweils zwei Funktionsbausteine zum Auslesen von Variablen aus der XML-Datei:

- FB_XmlSrvRead
- FB_XmlSrvReadByName

und zwei Funktionsbausteine zum Schreiben von SPS-Variablen in die XML-Datei:

- FB_XmlSrvWrite
- FB_XmlSrvWriteByName

Die erste Variante (FB_XMLSrvRead, FB_XMLSrvWrite) verwendet die Adresse und die Größe der Variable in der SPS, um die Variable zu spezifizieren. Die zweite Variante (FB_XMLSrvReadByName, FB_XMLSrvWriteByName) verwendet den Symbolnamen, um die Variable zu spezifizieren. Die erste Variante ist performanter. Zusätzlich zu Adresse und Größe bzw. Symbolname muss der Pfad der XML-Datei und der Standort der Variablen im XML-Dokument den Funktionsbausteinen als Eingangsparameter übergeben werden.

Primitive Datentypen

Folgende Primitive Datentypen werden unterstützt:

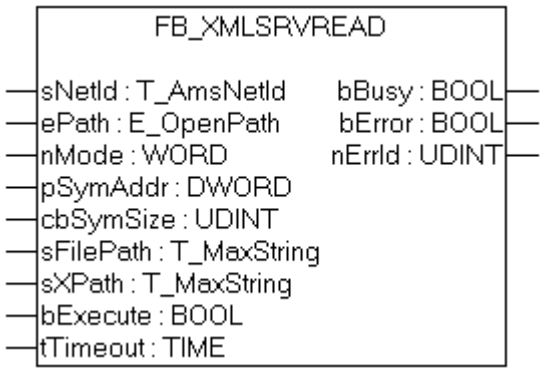
Datentyp	Beispiel in der SPS	Beispiel in XML
UDINT, DINT, UINT, INT, USINT, SINT, DWORD, WORD, BYTE	value1 : DINT := -1; value2 : UDINT := 65535;	<dataentry> <MAIN.value1>-1</MAIN.value1> <MAIN.value2>65535</MAIN.value2> </dataentry>
LREAL, REAL	value1 : LREAL = 1.2;	<dataentry> <MAIN.value1>1.2</MAIN.value1> </dataentry>
STRING	str1 : STRING = 'hallo';	<dataentry> <MAIN.str1>hallo</MAIN.str1> </dataentry>
TIME, DATE, TOD,DT	date1:DATE :=D#2005-05-04; (* Time-Typen werden in der XML-Datei als DWORD gespeichert*)	<dataentry> <MAIN.date1>1115164800</MAIN.date1> </dataentry>
BOOL	bool1:BOOL := TRUE; bool2:BOOL := FALSE;	<dataentry> <MAIN.bool1>true</MAIN.bool1> <MAIN.bool2>false</MAIN.bool2> </dataentry>

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

4.2 Funktionsbausteine

4.2.1 FB_XmlSrvRead



Mit dem Funktionsbaustein FB_XmlSrvRead kann eine SPS-Variable mit Daten aus einer XML-Datei initialisiert werden. Die Eingangsvariable sXPath muss dabei auf einen gültigen Knoten in der mit sFilePath angegebenen XML-Datei zeigen. Das Symbol, das initialisiert werden soll, wird anhand der übergebenen Symboladresse und Symbolgröße eindeutig identifiziert.

VAR_INPUT

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    ePath       : E_OpenPath := PATH_GENERIC;
    nMode       : WORD := XMLSRV_SKIPMISSING;
    pSymAddr    : DWORD;
    cbSymSize   : UDINT;
    sFilePath   : T_MaxString;
    sXPath      : T_MaxString;
    bExecute    : BOOL;
    tTimeout    : TIME := T#60s;
END_VAR
```

sNetId: String mit der Netzwerkadresse des TwinCAT 3 XML Servers. Für den lokalen Rechner (default) kann auch ein Leerstring angegeben werden.

ePath: Über diesen Eingang kann ein TwinCAT-Systempfad auf dem Zielgerät zum Öffnen der Datei angewählt werden.

nMode: Über diesen Eingang ist das Verhalten beeinflussbar, wie die XML-Datei ausgewertet wird. Für den Befehl XmlSrvRead wird zurzeit nur der Modus XMLSRV_SKIPMISSING unterstützt.

pSymAddr: Adresse der SPS-Variablen, die mit den Daten aus der XML-Datei beschrieben werden soll.

cbSymSize: Größe der SPS-Variablen, die mit den Daten aus der XML-Datei beschrieben werden soll.

sFilePath: Enthält den Pfad- und Dateinamen der zu öffnenden Datei. Der Pfad kann nur auf das lokale Filesystem des Rechners zeigen! Das bedeutet, Netzwerkpfade können hier nicht angegeben werden!

sXPath: Enthält die Adresse des Tags im XML-Dokument, aus der die Daten geschrieben werden soll. Die Adresse muss eine gültige XPath-Anweisung sein. Der Name des Tags muss dabei nicht dem Namen des Symbols entsprechen.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
END_VAR
```

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt so lange gesetzt, bis eine Rückmeldung erfolgt.

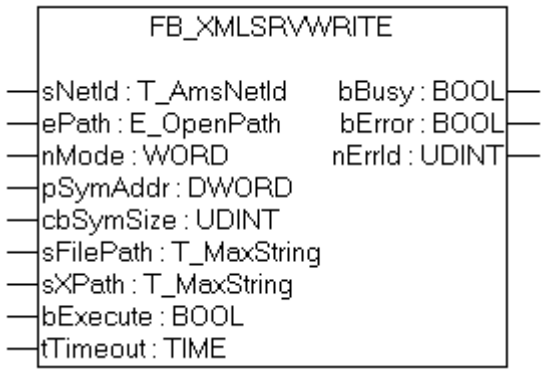
bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

nErrId: Liefert bei einem gesetzten bError-Ausgang die TC3 XML Server Fehlernummer [▶ 33].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

4.2.2 FB_XmlSrvWrite



Mit dem Funktionsbaustein FB_XmlSrvWrite kann der Wert einer SPS-Variablen in eine XML-Datei geschrieben werden. Die Eingangsvariable sXPath muss dabei auf einen gültigen Knoten in der mit sFilePath angegebenen XML-Datei zeigen. Das Symbol, das geschrieben werden soll, wird anhand der übergebenen Symboladresse und Symbolgröße eindeutig identifiziert.

VAR_INPUT

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    ePath       : E_OpenPath := PATH_GENERIC;
    nMode       : WORD := XMLSRV_SKIPMISSING;
    pSymAddr    : DWORD;
    cbSymSize   : UDINT;
    sFilePath   : T_MaxString;
    sXPath      : T_MaxString;
    bExecute    : BOOL;
    tTimeout    : TIME := T#60s;
END_VAR
```

sNetId: String mit der Netzwerkadresse des TwinCAT 3 Xml Servers. Für den lokalen Rechner (default) kann auch ein Leerstring angegeben werden.

ePath: Über diesen Eingang kann ein TwinCAT-Systempfad auf dem Zielgerät zum Öffnen der Datei angewählt werden.

nMode: Über diesen Eingang ist das Verhalten beeinflussbar, mit welchen Daten die XML-Datei beschrieben wird. Für den Befehl XmlSrvWrite gibt es den Modus XMLSRV_SKIPMISSING und XMLSRV_ADDMISSING. Im Modus XMLSRV_SKIPMISSING werden nur die Sub-Elemente eines SPS-Symbols in die XML-Datei geschrieben, die bereits vorher in der XML-Datei existierten. Im Modus XMLSRV_ADDMISSING werden in der XML-Datei fehlende Subelement der XML-Datei hinzugefügt. Ab Produktversion 3.2.31.0 ist es möglich mit den Parametern XMLSRV_SERIALIZESYMCOMMENT und XMLSRV_SERIALIZETYPECOMMENT die Kommentare aus dem Deklarationsbereich mit in die XML Datei zu schreiben.

pSymAddr: Adresse der SPS-Variablen, die in die XML-Datei geschrieben werden soll.

cbSymSize: Größe der SPS-Variablen, die in die XML-Datei geschrieben werden soll.

sFilePath: Enthält den Pfad- und Dateinamen der zu öffnenden Datei. Der Pfad kann nur auf das lokale Filesystem des Rechners zeigen! Das bedeutet, Netzwerkpfade können hier nicht angegeben werden!

sXPath: Enthält die Adresse des Tags im XML-Dokument, aus der die Daten geschrieben werden soll. Die Adresse muss eine gültige XPath-Anweisung sein. Der Name des Tags muss dabei nicht dem Namen des Symbols entsprechen.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
END_VAR
```

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt so lange gesetzt, bis eine Rückmeldung erfolgt.

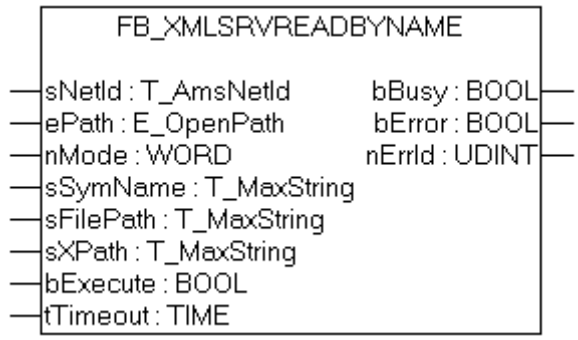
bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

nErrId: Liefert bei einem gesetzten bError-Ausgang die TC3 XML Server Fehlernummer [▶ 33].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

4.2.3 FB_XmlSrvReadByName



Mit dem Funktionsbaustein FB_XmlSrvReadByName kann eine SPS-Variable mit Daten aus einer XML-Datei initialisiert werden. Die Eingangsvariable sXPath muss dabei auf einen gültigen Knoten in der mit sFilePath angegebenen XML-Datei zeigen. Das Symbol, das initialisiert werden soll, wird anhand des Symbolnamens eindeutig identifiziert.

VAR_INPUT

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    ePath       : E_OpenPath := PATH_GENERIC;
    nMode       : WORD := XMLSRV_SKIPMISSING;
    sSymName    : T_MaxString;
    sFilePath   : T_MaxString;
    sXPath      : T_MaxString;
    bExecute    : BOOL;
    tTimeout    : TIME := T#60s;
END_VAR
```

sNetId: String mit der Netzwerkadresse des TwinCAT 3 XML Servers. Für den lokalen Rechner (default) kann auch ein Leerstring angegeben werden.

ePath: Über diesen Eingang kann ein TwinCAT-Systempfad auf dem Zielgerät zum Öffnen der Datei angewählt werden.

nMode: Über diesen Eingang ist das Verhalten beeinflussbar, wie die XML-Datei ausgewertet wird. Für den Befehl XmlSrvReadByName wird zurzeit nur der Modus XMLSRV_SKIPMISSING unterstützt.

sSymName: Name des SPS-Symbols, das mit den Daten aus der XML-Datei beschrieben werden soll.

sFilePath: Enthält den Pfad- und Dateinamen der zu öffnenden Datei. Der Pfad kann nur auf das lokale Filesystem des Rechners zeigen! Das bedeutet, Netzwerkpfade können hier nicht angegeben werden!

sXPath: Enthält die Adresse des Tags im XML-Dokument, aus der die Daten geschrieben werden soll. Die Adresse muss eine gültige XPath-Anweisung sein. Der Name des Tags muss dabei nicht dem Namen des Symbols entsprechen.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
END_VAR
```

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt so lange gesetzt, bis eine Rückmeldung erfolgt.

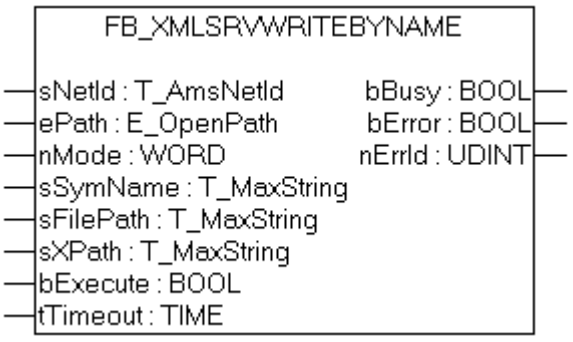
bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

nErrId: Liefert bei einem gesetzten bError-Ausgang die TC3 XML Server Fehlernummer [▶ 33].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

4.2.4 FB_XmlSrvWriteByName



Mit dem Funktionsbaustein FB_XmlSrvWriteByName kann der Wert einer SPS-Variablen in eine XML-Datei geschrieben werden. Die Eingangsvariable sXPath muss dabei auf einen gültigen Knoten in der mit sFilePath angegebenen XML-Datei zeigen. Das Symbol, das geschrieben werden soll, wird anhand des Symbolnamens eindeutig identifiziert.

VAR_INPUT

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    ePath       : E_OpenPath := PATH_GENERIC;
    nMode       : WORD := XMLSRV_SKIPMISSING;
    sSymName    : T_MaxString;
    sFilePath   : T_MaxString;
    sXPath      : T_MaxString;
    bExecute    : BOOL;
    tTimeout    : TIME := T#60s;
END_VAR
```

sNetId: String mit der Netzwerkadresse des TwinCAT 3 XML Servers. Für den lokalen Rechner (default) kann auch ein Leerstring angegeben werden.

ePath: Über diesen Eingang kann ein TwinCAT-Systempfad auf dem Zielgerät zum Öffnen der Datei angewählt werden.

nMode: Über diesen Eingang ist das Verhalten beeinflussbar, mit welchen Daten die XML-Datei beschrieben wird. Für den Befehl XmlSrvWriteByte gibt es den Modus XMLSRV_SKIPMISSING und XMLSRV_ADDMISSING. Im Modus XMLSRV_SKIPMISSING werden nur die Sub-Elemente eines SPS-Symbols in die XML-Datei geschrieben, die bereits vorher in der XML-Datei existierten. Im Modus XMLSRV_ADDMISSING werden in der XML-Datei fehlende Subelemente der XML-Datei hinzugefügt. Ab Produktversion 3.2.31.0 ist es möglich mit den Parametern XMLSRV_SERIALIZESYMCOMMENT und XMLSRV_SERIALIZETYPECOMMENT die Kommentare aus dem Deklarationsbereich mit in die XML Datei zu schreiben.

sSymName: Name des SPS-Symbols, das in die XML-Datei geschrieben werden soll.

sFilePath: Enthält den Pfad- und Dateinamen der zu öffnenden Datei. Der Pfad kann nur auf das lokale Filesystem des Rechners zeigen! Das bedeutet, Netzwerkpfade können hier nicht angegeben werden!

sXPath: Enthält die Adresse des Tags im XML-Dokument, aus der die Daten geschrieben werden soll. Die Adresse muss eine gültige XPath-Anweisung sein. Der Name des Tags muss dabei nicht dem Namen des Symbols entsprechen.

bExecute: Über eine positive Flanke an diesem Eingang wird der Baustein aktiviert.

tTimeout: Maximale Zeit, die bei der Ausführung des Funktionsbausteins nicht überschritten werden darf.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy   : BOOL;
    bError  : BOOL;
    nErrId  : UDINT;
END_VAR
```

bBusy: Dieser Ausgang wird bei der Aktivierung des Funktionsbausteins gesetzt und bleibt so lange gesetzt, bis eine Rückmeldung erfolgt.

bError: Dieser Ausgang wird, nachdem der bBusy-Ausgang zurückgesetzt wurde, gesetzt, sollte ein Fehler bei der Übertragung des Kommandos erfolgen.

nErrId: Liefert bei einem gesetzten bError-Ausgang die TC3 XML Server Fehlernummer [▶ 33].

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

4.3 Globale Konstanten

4.3.1 Globale Variablen

VAR_GLOBAL CONSTANT

```
VAR_GLOBAL CONSTANT

XMLSRV_AMSPORT :UINT :=10600;

XMLSRV_IGR_CLOSE :UDINT := 121;
XMLSRV_IGR_READ :UDINT := 122;
XMLSRV_IGR_WRITE :UDINT := 123;
XMLSRV_IGR_OPENREAD :UDINT := 124;
XMLSRV_IGR_OPENWRITE :UDINT := 125;

XMLSRV_SKIPMISSING :WORD := 0;
XMLSRV_ADDMISSING :WORD := 1; (*for write commands*)

XMLSRV_MAX_FRAGSIZE :UDINT := 16#40000;

XMLSRVERROR_INTERNAL :UDINT:= 16#8000;
XMLSRVERROR_NOTFOUND :UDINT:= 16#8001;
XMLSRVERROR_PARSERERROR :UDINT:= 16#8002;
XMLSRVERROR_INCOMPATIBLE :UDINT:= 16#8003;
XMLSRVERROR_NOMEMORY :UDINT:= 16#8004;
XMLSRVERROR_ADDNODE :UDINT:= 16#8005;
XMLSRVERROR_INVALIDXPATH :UDINT:= 16#8006;

END_VAR
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

4.3.2 Bibliotheksversion

Alle Bibliotheken haben eine bestimmte Version. Diese Version ist u. a. im SPS-Bibliotheks-Repository zu sehen. Eine globale Konstante enthält die Information über die Bibliotheksversion:

Global_Version

```
VAR_GLOBAL CONSTANT
    stLibVersion_Tc2_TcXmlDataSrv : ST_LibVersion;
END_VAR
```

stLibVersion_Tc2_TcXmlDataSrv: Versionsnummer der Tc2_TcXmlDataSry-Bibliothek (Typ: ST_LibVersion)

Um zu sehen, ob die Version, die Sie haben auch die Version ist, die Sie brauchen, benutzen Sie die Funktion F_CmpLibVersion (definiert in der Tc2_System-Bibliothek).

Version vergleichen

Alle anderen Möglichkeiten Bibliotheksversionen zu vergleichen, die Sie von TwinCAT 2 kennen, sind veraltet.

5 Beispiele

Auf den folgenden Seiten finden Sie Beispiele (Samples), die den Umgang mit dem TC3 XML Server zeigen. Die Beispiele sind fortlaufend nummeriert und können als PLC-Projekt hier heruntergeladen werden: https://infosys.beckhoff.com/content/1031/TF6421_Tc3_XML_Server/Resources/1635013643.zip

- Getting Started [▶ 24]
  Hier wird der generelle Umgang mit dem TwinCAT XML Data Server sowie das Arbeiten mit Strukturen und Arrays und der Zugriff auf einzelne Knotenpunkte behandelt.
- Die Funktionsbausteine [▶ 25]
  In vier Beispielen werden der Einsatz und die Konfiguration der Funktionsbausteine der SPS-Bibliothek TcXmlDataSrv.Lib vorgestellt. (Beispiel 1-4)
- Weiterführende Beispiele [▶ 28]
  Dieses Dokument enthält weiterführende Beispiele, die die einmalige Initialisierung bei SPS-Start oder das zyklische Schreiben darstellen. (Beispiel 5-6)
- Produktionsbeispiel [▶ 30]
  Das Produktionsbeispiel zeigt die Abarbeitung eines kleinen Produktionsauftrags. Es verwendet FB_XmlSrvReadByName und FB_XmlSrvWriteByName. (Beispiel 7)

5.1 Getting Started

Im Folgenden wird die grundlegende Arbeitsweise mit dem TC3 XML Server anhand kleiner Beispiele erklärt.

In der SPS ist die globale Variable .Var1 vom Typen DINT definiert. Diese soll in einer XML-Datei wie folgt abgespeichert werden:

```
<dataentry>
   <Var1>10</Var1>
</dataentry>
```

Dazu müssen die Inputvariablen des Funktionsbausteines FB_XmlSrvWrite wie folgt gesetzt werden:

```
fbRead.pSymAddr    := ADR(value1);
fbRead.cbSymSize   := SIZEOF(value1);
fbRead.sFilePath   :='C:\Test.xml'; (*Pfad zur XML-Datei*)
fbRead.sXPath      := '/dataentry/Var1';
```

Das Root-Element Name der Variablen in der XML-Datei ist frei wählbar. Es muss lediglich der Pfad in der Eingangsvariable sXPath angepasst. In einer XML-Datei können auch mehrere Variablen definiert sein:

```
<dataentry>
   <Var1>10</Var1>
   <Var2>100</Var2>
   <Var3>
      <a>100</a>
      <b>10</b>
   </Var3>
</dataentry>
```

Um z.B. auf das Symbol .Var3.a zuzugreifen, muss sXpath:= '/dataentry/Var3.a' gesetzt werden.

Strukturen

Strukturen in der XML-Datei haben den gleichen hierarchischen Aufbau wie in der SPS. Es gibt jedoch die Möglichkeit, einzelne Unterelemente in der XML-Datei zu überspringen: Die Unterelemente der Struktur müssen die gleichen Namen besitzen wie in der SPS, ansonsten werden sie übersprungen. Wenn Unterelemente in der XML-Datei nicht in den korrekten Datentypen umgewandelt werden können, werden auch sie übersprungen.

Beispiel:

In der SPS ist die globale Variable .Var2 vom Typ ST_MYSTRUCT definiert:

```
TYPE ST_MYSTRUCT:
STRUCT
    a: UINT;
```

```
    b: DINT;
    c: LREAL;
    d: STRING;
END_STRUCT
END_TYPE
```

Die XML-Datei könnte dann wie folgt aussehen:

```
<variables>
   <Var1>10</Var1>
   <Var2>     <!-- sXPath := '/variables/Var2' -->
      <a>100</a>
      <b>-10</b>
      <c>1.2</c>
      <d>Hallo</d>
   </Var2>
</variables>
```

In diesem Fall sind alle Unterelemente vollständig und korrekt definiert, sodass die Variable vollständig initialisiert wird. Im folgenden Beispiel hingegen wird nur das Unterelement c serialisiert:

```
<variables>
   <Var1>10</Var1>
   <Variable2> <!-- sXPath := '/variables/Variable2' -->
      <Info>dies ist ein Test</Info>
      <a>-100</a>
      <c>1.2</c>
   </Variable2>
</variables>
```

Unterelement a kann nicht konvertiert werden, weil es negativ ist und ein UINT gefordert wird. Das Unterelement b fehlt komplett. Das Tag <Info> wird übersprungen, da es in der SPS-Datei nicht definiert ist.

Arrays

Um den Index von Arrays anzugeben, muss bei den einzelnen Array-Elementen das Attribut "Index" verwendet werden. Es können auch einzelne Array-Elemente weggelassen werden. Diese werden dann einfach übersprungen.

Beispiel:

In der SPS ist eine Variable .array1 vom Typen ARRAY[1..4] OF DINT definiert. Die XML sieht dann wie folgt aus:

```
<dataentry>
   <array1 index="1">10</array1>
   <array1 index="2">10</array1>
   <array1 index="3">10</array1>
   <array1 index="4">10</array1>
</dataentry>
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

5.2 Die Funktionsbausteine

Die folgenden Beispiele zeigen den Umgang mit den Funktionsbausteinen der Tc2_XmlDataSrv-Bibliothek. Das SPS-Projekt, das die Beispiele enthält, können Sie hier herunterladen: https://infosys.beckhoff.com/content/1031/TF6421_Tc3_XML_Server/Resources/1635013643.zip

Alle Beispiele arbeiten mit der Struktur ST_MYSTRUCT, die wiederum die Struktur ST_INNTERSTRUCT enthält. Die beiden Strukturen sind im Folgenden dargestellt:

Die Strukturen

```
TYPE ST_MYSTRUCT:
STRUCT
    fReal     : REAL;
    bBool     : ARRAY [0..2] OF BOOL;
```

```
    stInner   : ST_INNTERSTRUCT;
END_STRUCT
END_TYPE

TYPE ST_INNTERSTRUCT:
STRUCT
    nInteger  : INT;
    sString   : STRING;
END_STRUCT
END_TYPE
```

Sample 1: Schreibvorgang mit FB_XmlSrvWrite

Im ersten Schritt soll die Struktur ST_MyStruct in eine XML-Datei geschrieben werden. Der Modus wird auf XMLSRV_ADMISSING gesetzt, sodass die XML-Datei automatisch erstellt und die Struktur darin angelegt wird. Ordner werden nicht automatisch angelegt! Dieses Vorgehen empfiehlt sich bei größeren Strukturen auch dann, wenn die Datei später nur ausgelesen werden soll. So muss die XML-Datei nicht manuell angelegt werden, Fehler werden vermieden.

```
(* Sample1 creates an XML-file under the path C:\Test.xml and writes value1 to it.
   FUNCTIONBLOCK: FB_XmlSrvWrite *)

PROGRAM Sample1
VAR
   value1         : ST_MyStruct;
   fbXmlSrvWrite  : FB_XmlSrvWrite;
   bExecute       : BOOL;
   sFilePath      : T_MaxString := 'C:\Test.xml'; (* CE: '\Hard Disk\Test.xml' *)
   sXPath         : T_MaxString := '/dataentry/MAIN.value1';
END_VAR

fbXmlSrvWrite(
 nMode       := XMLSRV_ADDMISSING,
 pSymAddr    := ADR(value1),
 cbSymSize   := SIZEOF(value1),
 sFilePath   := sFilePath,
 sXPath      := sXPath,
 bExecute    := bExecute
);
bExecute:= TRUE;
```

Sample 2: Schreibvorgang mit FB_XmlSrvWriteByName

Sample 2 führt zum gleichen Ergebnis wie Sample 1, verwendet aber den FB_XmlSrvWriteByName-Baustein. Sample 1 ist jedoch performanter.

```
(* Sample2 creates an XML-file under the path C:\Test.xml and writes value1 to it.
   FUNCTIONBLOCK: FB_XmlSrvWriteByName *)

PROGRAM Sample2
VAR
   value1         : ST_MyStruct;
   fbXmlSrvWrite  : FB_XmlSrvWriteByName;
   bExecute       : BOOL;
   sSymName       : T_MaxString := 'Sample2.value1';
   sFilePath      : T_MaxString := 'C:\Test.xml'; (* CE: '\Hard Disk\Test.xml' *)
   sXPath         : T_MaxString := '/dataentry/MAIN.value1';
END_VAR

fbXmlSrvWrite(
 nMode      := XMLSRV_ADDMISSING,
 sSymName   := sSymName,
 sFilePath  := sFilePath,
 sXPath     := sXPath,
 bExecute   := bExecute
);
bExecute:= TRUE;
```

XML-Datei

Bei beiden Beispielen wird die folgende XML-Datei *'Test.xml'* unter *C:* erstellt. Damit die XML-Datei bei beiden Varianten den gleichen Inhalt enthält, wird als sXPath der gleiche Pfad *'/dataentry/MAIN.value1'* gewählt, obwohl value1 nicht in der Main, sondern direkt in den entsprechenden Programmen liegt. sSymName (Sample 2) gibt hingegen den Ort der Variablen in TwinCAT an: *'Sample2.value1'!*

```
<dataentry>
  <MAIN.value1>
    <fReal>0</fReal>
    <bBool index="0">false</bBool>
    <bBool index="1">false</bBool>
    <bBool index="2">false</bBool>
    <stInner>
      <nInteger>0</nInteger>
      <sString></sString>
    </stInner>
  </MAIN.value1>
</dataentry>
```

Sample 3: Lesevorgang mit FB_XmlSrvRead

Im Folgenden soll die Struktur aus der in Sample 1 bzw. Sample 2 erstellten XML-Datei wieder eingelesen werden. Sample 3 nutzt hierzu den FB_XmlSrvRead-Baustein.

```
(* Sample3 reads an XML-file (C:\Test.xml)
   FUNCTIONBLOCK: FB_XmlSrvRead *)

PROGRAM Sample3

VAR
   value1        : ST_MyStruct;
   fbXmlSrvRead  : FB_XmlSrvRead;
   bExecute      : BOOL;
   sFilePath     : T_MaxString := 'C:\Test.xml'; (* CE: '\Hard Disk\Test.xml' *)
   sXPath        : T_MaxString := '/dataentry/MAIN.value1';
END_VAR

fbXmlSrvRead(
 pSymAddr   := ADR(value1),
 cbSymSize  := SIZEOF(value1),
 sFilePath  := sFilePath,
 sXPath     := sXPath,
 bExecute   := bExecute
);
bExecute:= TRUE;
```

Sample 4: Lesevorgang mit FB_XmlSrvReadByName

Sample 4 zeigt den Lesevorgang unter Verwendung des FB_XmlSrvReadByName-Bausteins.

```
(* Sample4 reads an XML-file (C:\Test.xml)
   FUNCTIONBLOCK: FB_XmlSrvReadByName *)

PROGRAM Sample4
VAR
   value1         : ST_MyStruct;
   fbXmlSrvRead   : FB_XmlSrvReadByName;
   bExecute       : BOOL;
   sSymName       : T_MaxString := 'Sample4.value1';
   sFilePath      : T_MaxString := 'C:\Test.xml'; (* CE: '\Hard Disk\Test.xml' *)
   sXPath         : T_MaxString := '/dataentry/MAIN.value1';
END_VAR

fbXmlSrvRead(
 sSymName    := sSymName,
 sFilePath   := sFilePath,
 sXPath      := sXPath,
 bExecute    := bExecute
);
bExecute:= TRUE;
```

Bei Sample 4 ist erneut (wie bei Sample 2) zu beachten, dass sich sSymName und sXPath unterscheiden: sXPath gibt den Pfad innerhalb der XML-Datei an. Dieser wurde in Sample 1 bzw. Sample 2 festgelegt. sSymName hingegen gibt den Symbolnamen der TwinCAT-Variablen an und lautet deshalb *'Sample4.value1'*.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

5.3 Weiterführende Beispiele

Die folgenden Beispiele zeigen verschiedene Anwendungsarten des TC3 XML Servers. Das SPS-Projekt, das die Beispiele enthält, können Sie hier herunterladen: https://infosys.beckhoff.com/content/1031/TF6421_Tc3_XML_Server/Resources/1635013643.zip

Beispiel 5 zeigt eine einmalige Initialisierung bei Programmstart, Beispiel 6 zyklische und ereignisgesteuerte Schreibvorgänge. Beide Beispiele arbeiten mit der schon aus den Beispielen 1-4 bekannten Struktur ST_MYSTRUCT, die wiederum die Struktur ST_INNTERSTRUCT enthält. Die beiden Strukturen sind im Folgenden dargestellt:

Die Strukturen

```
TYPE ST_MYSTRUCT:
STRUCT
    fReal    : REAL;
    bBool    : ARRAY [0..2] OF BOOL;
    stInner  : ST_INNTERSTRUCT;
END_STRUCT
END_TYPE

TYPE ST_INNTERSTRUCT:
STRUCT
    nInteger : INT;
    sString  : STRING;
END_STRUCT
END_TYPE
```

Sample 5: Einmalige Initalisierung bei Programmstart

Sample 5 zeigt die einmalige Initialisierung von value1 bei Programmstart. Zum Einsatz kommt der Funktionsbaustein FB_XmlSrvRead.

```
(* Sample5 reads and initializes value1 when the PLC is started FUNCTIONBLOCK: FB_XmlSrvRead *)

PROGRAM Sample5
VAR
   value1        : ST_MyStruct;
   fbXmlSrvRead  : FB_XmlSrvRead;
   bExecute      : BOOL;
   sFilePath     : T_MaxString := 'C:\Test.xml'; (* CE: '\Hard Disk\Test.xml' *)
   sXPath        : T_MaxString := '/dataentry/MAIN.value1';
   nState        : INT := 0;
END_VAR

CASE nState OF
0: (* initialize *)
   fbXmlSrvRead(
      pSymAddr   := ADR(value1),
      cbSymSize  := SIZEOF(value1),
      sFilePath  := sFilePath,
      sXPath     := sXPath,
      bExecute   := bExecute
   );
   fbXmlSrvRead(bExecute:= TRUE);
   nState:= 1;

1: (* wait for read operation *)
   fbXmlSrvRead(bExecute:= FALSE);
   IF NOT fbXmlSrvRead.bBusy AND NOT fbXmlSrvRead.bError THEN
      nState:= 2;
   ELSIF fbXmlSrvRead.bError THEN
      nState:= 100;
   END_IF

2: (* operations *)
 ;

100:(* errorState *)
 ;

END_CASE
```

Sample 6: Zyklisches und ereignisgesteuertes Schreiben

Das folgende Beispiel erzeugt alle 20 Sekunden eine neue XML-Datei und beschreibt diese mit der bekannten Struktur. Der Dateiname wird immer aus dem aktuellen Windows-Datum, der -Uhrzeit und einem String generiert. Außerdem kann der Schreibprozess durch Drücken eines Schalters (bzw. Setzen der Schaltervariable bButton) gestartet werden. Sollte der Schreibprozess in einer Sekunde mehrfach ausgelöst werden, wird die aktuelle Datei überschrieben.

```
(* Sample6: Every 20s value1 will be written into a new XML-File named after the current date and
 time. Furthermore you can activate the printing procedure by pressing a button (or setting the
 corresponding variable *)


PROGRAM Sample6
VAR
   value1          : ST_MyStruct;
   fbXmlSrvWrite   : FB_XmlSrvWrite;

   sFileFolder     : T_MaxString :='C:\'; (* CE: '\Hard Disk\' *)
   sFileName       : T_MaxString:= '_test.xml';
   sFilePathWrite  : T_MaxString
   (*sFilePathWrite = sFileFolder + time + sFileName*)

   sXPathWrite     : T_MaxString :='/dataentry/MAIN.value1';
   ntGetTime       : NT_GetTime;
   stMyTimestruct  : TIMESTRUCT;
   iState          : INT := 1;
   bTwentySec      : BOOL:= FALSE;
   bButton         : BOOL:= FALSE;
   bTwentySecOver  : BOOL;
   triggerWrite    : R_TRIG;
   triggerButton   : R_TRIG;
END_VAR


triggerButton(CLK:= bButton);

CASE iState OF
0: (* idle state *)
 ;

1: (* initialize *)
   fbXmlSrvWrite(nMode:=XMLSRV_ADDMISSING, pSymAddr:= ADR(value1),
   cbSymSize:= SIZEOF(value1));
   ntGetTime(START:= TRUE, TIMESTR=>stMyTimestruct); (* get Windows time *)
   IF NOT ntGetTime.BUSY AND NOT ntGetTime.ERR THEN
      iState:= 2;
   ELSIF ntGetTime.ERR THEN
      iState:= 100;
 END_IF

2: (* working state *)
   (* change some values - replace with production-process *)
   value1.stInner.nInteger:= value1.stInner.nInteger + 1;
   IF value1.stInner.nInteger = 32767 THEN
      value1.stInner.nInteger:= 0;
   END_IF(* get Windows time *)
   ntGetTime(START:= FALSE);
   IF NOT ntGetTime.BUSY AND NOT ntGetTime.ERR THEN
      ntGetTime(START:= TRUE, TIMESTR=>stMyTimestruct);
   ELSIF ntGetTime.ERR THEN
      iState:= 100;
   END_IF


   (* check if 20s have passed*)
   IF stMyTimestruct.wSecond = 0 OR stMyTimestruct.wSecond = 20
                   OR stMyTimeStruct.wSecond = 40 THEN
      bTwentySecOver:= TRUE;
      ELSE
      bTwentySecOver:= FALSE;
   END_IF


   (* if 20s have passed => trigger writing-process *)
   triggerWrite(CLK:=bTwentySecOver);
   IF (triggerWrite.Q OR triggerButton.Q) AND NOT fbXmlSrvWrite.bBusy AND NOT fbXmlSrvWrite.bError T
HEN
      (* create filename *)
```

```
      sFilePathWrite:= CONCAT(sFileFolder, SYSTEMTIME_TO_STRING(stMyTimestruct)); (* set folder + ti
me *)
      sFilePathWrite:= DELETE(STR:= sFilePathWrite, LEN:= 4 , POS:= LEN(STR:=sFilePathWrite)-3); (*
delete milliseconds *)
      sFilePathWrite:= REPLACE(STR1:= sFilePathWrite , STR2:= '.' , L:= 1,
      P:= LEN(STR:=sFilePathWrite)-2); (* replace colon with point *)
      sFilePathWrite:= REPLACE(STR1:= sFilePathWrite , STR2:= '.' , L:= 1,
      P:= LEN(STR:=sFilePathWrite)-5); (* replace colon with point *)
      sFilePathWrite:= CONCAT(sFilePathWrite, sFileName); (* add filename (default: test) *)


      (*change value 1*)
      value1.stInner.sString := sFilePathWrite;
      value1.stInner.nInteger := stMyTimestruct.wSecond;

      (* write *)
      fbXmlSrvWrite(sFilePath:=sFilePathWrite, sXPath:=sXPathWrite, bExecute:= TRUE);

   ELSIF fbXmlSrvWrite.bError THEN
      iState:= 100;
   END_IF

   (* reset fbXmlSrvWrite *)
   IF fbXmlSrvWrite.bBusy AND NOT tGetTime.ERR THEN
      fbXmlSrvWrite(bExecute:= FALSE);
   ELSIF ntGetTime.ERR THEN
      iState:= 100;
   END_IF

100: (* error state*)
 ;

END_CASE
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

5.4 Produktionsbeispiel

Sample 7 (Produktionsbeispiel)

Hier finden Sie ein Beispiel für einen Produktionsauftrag. Dabei werden Produktionsdaten wie Bauteillänge, Bauteilbreite etc. in einer Struktur zur Initialisierung ausgelesen (XML Read), die Produktion mit einer entsprechenden Stückzahl durchgeführt und der Produktionsauftrag mit einem Eintrag in der XML-Datei (Write) beendet. Um das Programm zu starten, muss zuvor das XML File an die dem Datei-Pfad entsprechende Stelle gespeichert werden und im SPS-Programm die Variable bStart auf TRUE gesetzt werden.

Das SPS-Projekt, das die Beispiele enthält, können Sie hier herunterladen: https://infosys.beckhoff.com/content/1031/TF6421_Tc3_XML_Server/Resources/1635013643.zip

Variablendeklaration

```
PROGRAM Sample7
VAR
   fbXmlSrvReadByName   : FB_XmlSrvReadByName;
   fbXmlSrvWriteByName  : FB_XmlSrvWriteByName;
   value                : ST_MyProductionStruct;
   state                : INT := 0;
   R_Edge               : R_TRIG;
   bStart               : BOOL;
   bError               : BOOL;
   nErrId               : UDINT;
END_VAR
```

Struktur ST_MyProductionStruct

```
TYPE ST_MyProductionStruct :
STRUCT
   rLength    : REAL;
   rWidth     : REAL;
   rHeight    : REAL;
   iQuantity  : INT;
   iCounter   : INT;
   bReady     : BOOL;
   stInfo     : STRING;
END_STRUCT
END_TYPE
```

SPS-Programm

```
(* The production data is read, the production is carried out and, according to the
 quantity and the production order, completed with an entry in the XML file.
 To start the program the XML file needs to be stored in the corresponding folder (sFilePath) and th
e variable bStart needs to be set TRUE in the PLC program. *)

R_Edge (CLK := bStart);
IF R_Edge.Q THEN
   state := 1;
END_IF

CASE state OF
0: (* idle state *)
   ;

1: (* init state *)
fbXmlSrvReadByName( sNetId := '',
            sSymName := 'Sample7.value',
            sFilePath := 'C:\Production1.xml',
            sXPath := '/dataentry/MAIN.value',
            bExecute := TRUE,
            tTimeout := t#10s,
            bError => bError,
            nErrId => nErrId);
state := 2;

2:
fbXmlSrvReadByName(bExecute := FALSE);
IF NOT fbXmlSrvReadByName.bBusy AND NOT fbXmlSrvReadByName.bError THEN
   state := 3;
ELSIF fbXmlSrvReadByName.bError THEN
   state := 100;
END_IF

3: (* working state *)
IF value.bReady = TRUE THEN
   value.stInfo := 'The order was already processed!';
   (* replace your production XML file! *)
   state := 4;
   RETURN;
END_IF

(* call production program with
 new length, width and height here *)
 value.iCounter := value.iCounter + 1;
 IF value.iCounter = value.iQuantity THEN
    value.bReady := TRUE;
    state := 4;
END_IF

4: (* documentation state *)
fbXmlSrvWriteByName( sNetId := '',
             nMode := XMLSRV_SKIPMISSING,
             sSymName := 'Sample7.value',
             sFilePath := 'C:\Production1.xml',
             sXPath := '/dataentry/MAIN.value',
             bExecute := TRUE,
             tTimeout := t#10s,
             bError => bError,
             nErrId => nErrId);
 state := 5;

5:
fbXmlSrvWriteByName(bExecute := FALSE);
```

```
IF NOT fbXmlSrvWriteByName.bBusy AND NOT fbXmlSrvWriteByName.bError THEN
   state := 0;
ELSIF fbXmlSrvWriteByName.bError THEN
   state := 100;
END_IF

100:(* error state *)
   ;
```

XML-Datei

```
<dataentry>
  <MAIN.value>
    <rLength>65.85</rLength>
    <rWidth>30</rWidth>
    <rHeight>2.5</rHeight>
    <iQuantity>500</iQuantity>
    <iCounter>0</iCounter>
    <bReady>false</bReady>
    <stInfo></stInfo>
  </MAIN.value>
</dataentry>
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v3.1 Build 4011	PC oder CX (x86, x64, ARM)	Tc2_XmlDataSrv

6 Anhang

6.1 Übersicht der Fehlercodes des TC3 XML Servers

Offset + Fehlercode	Bereich	Beschreibung
0x00000000 + TwinCAT Systemfehler-Codes [▶ 33]	0x00000000-0x00007800	TwinCAT Systemfehler (ADS-Fehlercodes inklusive)
0x00008000 + Interne TC3 XML Server Fehler [▶ 38]	0x00008000-0x000080FF	Interne Fehler des TC3 XML Servers

6.2 ADS Return Codes

Gruppierung der Fehlercodes:
Globale Fehlercodes: 0x0000 [▶ 33]... (0x9811_0000 ...)
Router Fehlercodes: 0x0500 [▶ 34]... (0x9811_0500 ...)
Allgemeine ADS Fehler: 0x0700 [▶ 35]... (0x9811_0700 ...)
RTime Fehlercodes: 0x1000 [▶ 37]... (0x9811_1000 ...)

Globale Fehlercodes

Hex	Dec	HRESULT	Name	Beschreibung
0x0	0	0x98110000	ERR_NOERROR	Kein Fehler.
0x1	1	0x98110001	ERR_INTERNAL	Interner Fehler.
0x2	2	0x98110002	ERR_NORTIME	Keine Echtzeit.
0x3	3	0x98110003	ERR_ALLOCLOCKEDMEM	Zuweisung gesperrt - Speicherfehler.
0x4	4	0x98110004	ERR_INSERTMAILBOX	Postfach voll – Es konnte die ADS Nachricht nicht versendet werden. Reduzieren der Anzahl der ADS Nachrichten pro Zyklus bringt Abhilfe.
0x5	5	0x98110005	ERR_WRONGRECEIVEHMSG	Falsches HMSG.
0x6	6	0x98110006	ERR_TARGETPORTNOTFOUND	Ziel-Port nicht gefunden – ADS Server ist nicht gestartet oder erreichbar.
0x7	7	0x98110007	ERR_TARGETMACHINENOTFOUND	Zielrechner nicht gefunden – AMS Route wurde nicht gefunden.
0x8	8	0x98110008	ERR_UNKNOWNCMDID	Unbekannte Befehl-ID.
0x9	9	0x98110009	ERR_BADTASKID	Ungültige Task-ID.
0xA	10	0x9811000A	ERR_NOIO	Kein IO.
0xB	11	0x9811000B	ERR_UNKNOWNAMSCMD	Unbekannter AMS-Befehl.
0xC	12	0x9811000C	ERR_WIN32ERROR	Win32 Fehler.
0xD	13	0x9811000D	ERR_PORTNOTCONNECTED	Port nicht verbunden.
0xE	14	0x9811000E	ERR_INVALIDAMSLENGTH	Ungültige AMS-Länge.
0xF	15	0x9811000F	ERR_INVALIDAMSNETID	Ungültige AMS Net ID.
0x10	16	0x98110010	ERR_LOWINSTLEVEL	Installations-Level ist zu niedrig –TwinCAT 2 Lizenzfehler.
0x11	17	0x98110011	ERR_NODEBUGINTAVAILABLE	Kein Debugging verfügbar.
0x12	18	0x98110012	ERR_PORTDISABLED	Port deaktiviert – TwinCAT System Service nicht gestartet.
0x13	19	0x98110013	ERR_PORTALREADYCONNECTED	Port bereits verbunden.
0x14	20	0x98110014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 Fehler.
0x15	21	0x98110015	ERR_AMSSYNC_TIMEOUT	AMS Sync Timeout.
0x16	22	0x98110016	ERR_AMSSYNC_AMSERROR	AMS Sync Fehler.
0x17	23	0x98110017	ERR_AMSSYNC_NOINDEXINMAP	Keine Index-Map für AMS Sync vorhanden.
0x18	24	0x98110018	ERR_INVALIDAMSPORT	Ungültiger AMS-Port.
0x19	25	0x98110019	ERR_NOMEMORY	Kein Speicher.
0x1A	26	0x9811001A	ERR_TCPSEND	TCP Sendefehler.
0x1B	27	0x9811001B	ERR_HOSTUNREACHABLE	Host nicht erreichbar.
0x1C	28	0x9811001C	ERR_INVALIDAMSFRAGMENT	Ungültiges AMS Fragment.
0x1D	29	0x9811001D	ERR_TLSSEND	TLS Sendefehler – Secure ADS Verbindung fehlgeschlagen.
0x1E	30	0x9811001E	ERR_ACCESSDENIED	Zugriff Verweigert – Secure ADS Zugriff verweigert.

Router Fehlercodes

Hex	Dec	HRESULT	Name	Beschreibung
0x500	1280	0x98110500	ROUTERERR_NOLOCKEDMEMORY	Lockierter Speicher kann nicht zugewiesen werden.
0x501	1281	0x98110501	ROUTERERR_RESIZEMEMORY	Die Größe des Routerspeichers konnte nicht geändert werden.
0x502	1282	0x98110502	ROUTERERR_MAILBOXFULL	Das Postfach hat die maximale Anzahl der möglichen Meldungen erreicht.
0x503	1283	0x98110503	ROUTERERR_DEBUGBOXFULL	Das Debug Postfach hat die maximale Anzahl der möglichen Meldungen erreicht.
0x504	1284	0x98110504	ROUTERERR_UNKNOWNPORTTYPE	Der Porttyp ist unbekannt.
0x505	1285	0x98110505	ROUTERERR_NOTINITIALIZED	Router ist nicht initialisiert.
0x506	1286	0x98110506	ROUTERERR_PORTALREADYINUSE	Die Portnummer ist bereits vergeben.
0x507	1287	0x98110507	ROUTERERR_NOTREGISTERED	Der Port ist nicht registriert.
0x508	1288	0x98110508	ROUTERERR_NOMOREQUEUES	Die maximale Portanzahl ist erreicht.
0x509	1289	0x98110509	ROUTERERR_INVALIDPORT	Der Port ist ungültig.
0x50A	1290	0x9811050A	ROUTERERR_NOTACTIVATED	Der Router ist nicht aktiv.
0x50B	1291	0x9811050B	ROUTERERR_FRAGMENTBOXFULL	Das Postfach hat die maximale Anzahl für fragmentierte Nachrichten erreicht.
0x50C	1292	0x9811050C	ROUTERERR_FRAGMENTTIMEOUT	Fragment Timeout aufgetreten.
0x50D	1293	0x9811050D	ROUTERERR_TOBEREMOVED	Port wird entfernt.

Allgemeine ADS Fehlercodes

Hex	Dec	HRESULT	Name	Beschreibung
0x700	1792	0x98110700	ADSERR_DEVICE_ERROR	Allgemeiner Gerätefehler.
0x701	1793	0x98110701	ADSERR_DEVICE_SRVNOTSUPP	Service wird vom Server nicht unterstützt.
0x702	1794	0x98110702	ADSERR_DEVICE_INVALIDGRP	Ungültige Index-Gruppe.
0x703	1795	0x98110703	ADSERR_DEVICE_INVALIDOFFSET	Ungültiger Index-Offset.
0x704	1796	0x98110704	ADSERR_DEVICE_INVALIDACCESS	Lesen oder Schreiben nicht gestattet.
0x705	1797	0x98110705	ADSERR_DEVICE_INVALIDSIZE	Parametergröße nicht korrekt.
0x706	1798	0x98110706	ADSERR_DEVICE_INVALIDDATA	Ungültige Daten-Werte.
0x707	1799	0x98110707	ADSERR_DEVICE_NOTREADY	Gerät nicht betriebsbereit.
0x708	1800	0x98110708	ADSERR_DEVICE_BUSY	Gerät beschäftigt.
0x709	1801	0x98110709	ADSERR_DEVICE_INVALIDCONTEXT	Ungültiger Kontext vom Betriebssystem - Kann durch Verwendung von ADS Bausteinen in unterschiedlichen Tasks auftreten. Abhilfe kann die Multitasking-Syncronisation in der SPS geben.
0x70A	1802	0x9811070A	ADSERR_DEVICE_NOMEMORY	Nicht genügend Speicher.
0x70B	1803	0x9811070B	ADSERR_DEVICE_INVALIDPARM	Ungültige Parameter-Werte.
0x70C	1804	0x9811070C	ADSERR_DEVICE_NOTFOUND	Nicht gefunden (Dateien,...).
0x70D	1805	0x9811070D	ADSERR_DEVICE_SYNTAX	Syntax-Fehler in Datei oder Befehl.
0x70E	1806	0x9811070E	ADSERR_DEVICE_INCOMPATIBLE	Objekte stimmen nicht überein.
0x70F	1807	0x9811070F	ADSERR_DEVICE_EXISTS	Objekt ist bereits vorhanden.
0x710	1808	0x98110710	ADSERR_DEVICE_SYMBOLNOTFOUND	Symbol nicht gefunden.
0x711	1809	0x98110711	ADSERR_DEVICE_SYMBOLVERSIONINVALID	Symbol-Version ungültig – Kann durch einen Online-Change auftreten. Erzeuge einen neuen Handle.
0x712	1810	0x98110712	ADSERR_DEVICE_INVALIDSTATE	Gerät (Server) ist im ungültigen Zustand.
0x713	1811	0x98110713	ADSERR_DEVICE_TRANSMODENOTSUPP	AdsTransMode nicht unterstützt.
0x714	1812	0x98110714	ADSERR_DEVICE_NOTIFYHNDINVALID	Notification Handle ist ungültig.
0x715	1813	0x98110715	ADSERR_DEVICE_CLIENTUNKNOWN	Notification-Client nicht registriert.
0x716	1814	0x98110716	ADSERR_DEVICE_NOMOREHDLS	Keine weiteren Handles verfügbar.
0x717	1815	0x98110717	ADSERR_DEVICE_INVALIDWATCHSIZE	Größe der Notification zu groß.
0x718	1816	0x98110718	ADSERR_DEVICE_NOTINIT	Gerät nicht initialisiert.
0x719	1817	0x98110719	ADSERR_DEVICE_TIMEOUT	Gerät hat einen Timeout.
0x71A	1818	0x9811071A	ADSERR_DEVICE_NOINTERFACE	Interface Abfrage fehlgeschlagen.
0x71B	1819	0x9811071B	ADSERR_DEVICE_INVALIDINTERFACE	Falsches Interface angefordert.
0x71C	1820	0x9811071C	ADSERR_DEVICE_INVALIDCLSID	Class-ID ist ungültig.
0x71D	1821	0x9811071D	ADSERR_DEVICE_INVALIDOBJID	Object-ID ist ungültig.
0x71E	1822	0x9811071E	ADSERR_DEVICE_PENDING	Anforderung steht aus.
0x71F	1823	0x9811071F	ADSERR_DEVICE_ABORTED	Anforderung wird abgebrochen.
0x720	1824	0x98110720	ADSERR_DEVICE_WARNING	Signal-Warnung.
0x721	1825	0x98110721	ADSERR_DEVICE_INVALIDARRAYIDX	Ungültiger Array-Index.
0x722	1826	0x98110722	ADSERR_DEVICE_SYMBOLNOTACTIVE	Symbol nicht aktiv.
0x723	1827	0x98110723	ADSERR_DEVICE_ACCESSDENIED	Zugriff verweigert.
0x724	1828	0x98110724	ADSERR_DEVICE_LICENSENOTFOUND	Fehlende Lizenz.
0x725	1829	0x98110725	ADSERR_DEVICE_LICENSEEXPIRED	Lizenz abgelaufen.
0x726	1830	0x98110726	ADSERR_DEVICE_LICENSEEXCEEDED	Lizenz überschritten.
0x727	1831	0x98110727	ADSERR_DEVICE_LICENSEINVALID	Lizenz ungültig.
0x728	1832	0x98110728	ADSERR_DEVICE_LICENSESYSTEMID	Lizenzproblem: System-ID ist ungültig.
0x729	1833	0x98110729	ADSERR_DEVICE_LICENSENOTIMELIMIT	Lizenz nicht zeitlich begrenzt.
0x72A	1834	0x9811072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	Lizenzproblem: Zeitpunkt in der Zukunft.
0x72B	1835	0x9811072B	ADSERR_DEVICE_LICENSETIMETOLONG	Lizenz-Zeitraum zu lang.
0x72C	1836	0x9811072C	ADSERR_DEVICE_EXCEPTION	Exception beim Systemstart.
0x72D	1837	0x9811072D	ADSERR_DEVICE_LICENSEDUPLICATED	Lizenz-Datei zweimal gelesen.
0x72E	1838	0x9811072E	ADSERR_DEVICE_SIGNATUREINVALID	Ungültige Signatur.
0x72F	1839	0x9811072F	ADSERR_DEVICE_CERTIFICATEINVALID	Zertifikat ungültig.
0x730	1840	0x98110730	ADSERR_DEVICE_LICENSEOEMNOTFOUND	Public Key vom OEM nicht bekannt.
0x731	1841	0x98110731	ADSERR_DEVICE_LICENSERESTRICTED	Lizenz nicht gültig für diese System.ID.
0x732	1842	0x98110732	ADSERR_DEVICE_LICENSEDEMODENIED	Demo-Lizenz untersagt.
0x733	1843	0x98110733	ADSERR_DEVICE_INVALIDFNCID	Funktions-ID ungültig.
0x734	1844	0x98110734	ADSERR_DEVICE_OUTOFRANGE	Außerhalb des gültigen Bereiches.
0x735	1845	0x98110735	ADSERR_DEVICE_INVALIDALIGNMENT	Ungültiges Alignment.

Hex	Dec	HRESULT	Name	Beschreibung
0x736	1846	0x98110736	ADSERR_DEVICE_LICENSEPLATFORM	Ungültiger Plattform Level.
0x737	1847	0x98110737	ADSERR_DEVICE_FORWARD_PL	Kontext – Weiterleitung zum Passiv-Level.
0x738	1848	0x98110738	ADSERR_DEVICE_FORWARD_DL	Kontext – Weiterleitung zum Dispatch-Level.
0x739	1849	0x98110739	ADSERR_DEVICE_FORWARD_RT	Kontext – Weiterleitung zur Echtzeit.
0x740	1856	0x98110740	ADSERR_CLIENT_ERROR	Clientfehler.
0x741	1857	0x98110741	ADSERR_CLIENT_INVALIDPARM	Dienst enthält einen ungültigen Parameter.
0x742	1858	0x98110742	ADSERR_CLIENT_LISTEMPTY	Polling-Liste ist leer.
0x743	1859	0x98110743	ADSERR_CLIENT_VARUSED	Var-Verbindung bereits im Einsatz.
0x744	1860	0x98110744	ADSERR_CLIENT_DUPLINVOKEID	Die aufgerufene ID ist bereits in Benutzung.
0x745	1861	0x98110745	ADSERR_CLIENT_SYNCTIMEOUT	Timeout ist aufgetreten – Die Gegenstelle antwortet nicht im vorgegebenen ADS Timeout. Die Routeneinstellung der Gegenstelle kann falsch konfiguriert sein.
0x746	1862	0x98110746	ADSERR_CLIENT_W32ERROR	Fehler im Win32 Subsystem.
0x747	1863	0x98110747	ADSERR_CLIENT_TIMEOUTINVALID	Ungültiger Client Timeout-Wert.
0x748	1864	0x98110748	ADSERR_CLIENT_PORTNOTOPEN	Port nicht geöffnet.
0x749	1865	0x98110749	ADSERR_CLIENT_NOAMSADDR	Keine AMS Adresse.
0x750	1872	0x98110750	ADSERR_CLIENT_SYNCINTERNAL	Interner Fehler in Ads-Sync.
0x751	1873	0x98110751	ADSERR_CLIENT_ADDHASH	Überlauf der Hash-Tabelle.
0x752	1874	0x98110752	ADSERR_CLIENT_REMOVEHASH	Schlüssel in der Tabelle nicht gefunden.
0x753	1875	0x98110753	ADSERR_CLIENT_NOMORESYM	Keine Symbole im Cache.
0x754	1876	0x98110754	ADSERR_CLIENT_SYNCRESINVALID	Ungültige Antwort erhalten.
0x755	1877	0x98110755	ADSERR_CLIENT_SYNCPORTLOCKED	Sync Port ist verriegelt.
0x756	1878	0x98110756	ADSERR_CLIENT_REQUESTCANCELLED	Die Anfrage wurde abgebrochen.

RTime Fehlercodes

Hex	Dec	HRESULT	Name	Beschreibung
0x1000	4096	0x98111000	RTERR_INTERNAL	Interner Fehler im Echtzeit-System.
0x1001	4097	0x98111001	RTERR_BADTIMERPERIODS	Timer-Wert nicht gültig.
0x1002	4098	0x98111002	RTERR_INVALIDTASKPTR	Task-Pointer hat den ungültigen Wert 0 (null).
0x1003	4099	0x98111003	RTERR_INVALIDSTACKPTR	Stack-Pointer hat den ungültigen Wert 0 (null).
0x1004	4100	0x98111004	RTERR_PRIOEXISTS	Die Request Task Priority ist bereits vergeben.
0x1005	4101	0x98111005	RTERR_NOMORETCB	Kein freier TCB (Task Control Block) verfügbar. Maximale Anzahl von TCBs beträgt 64.
0x1006	4102	0x98111006	RTERR_NOMORESEMAS	Keine freien Semaphoren zur Verfügung. Maximale Anzahl der Semaphoren beträgt 64.
0x1007	4103	0x98111007	RTERR_NOMOREQUEUES	Kein freier Platz in der Warteschlange zur Verfügung. Maximale Anzahl der Plätze in der Warteschlange beträgt 64.
0x100D	4109	0x9811100D	RTERR_EXTIRQALREADYDEF	Ein externer Synchronisations-Interrupt wird bereits angewandt.
0x100E	4110	0x9811100E	RTERR_EXTIRQNOTDEF	Kein externer Sync-Interrupt angewandt.
0x100F	4111	0x9811100F	RTERR_EXTIRQINSTALLFAILED	Anwendung des externen Synchronisierungs-Interrupts ist fehlgeschlagen.
0x1010	4112	0x98111010	RTERR_IRQLNOTLESSOREQUAL	Aufruf einer Service-Funktion im falschen Kontext
0x1017	4119	0x98111017	RTERR_VMXNOTSUPPORTED	Intel VT-x Erweiterung wird nicht unterstützt.
0x1018	4120	0x98111018	RTERR_VMXDISABLED	Intel VT-x Erweiterung ist nicht aktiviert im BIOS.
0x1019	4121	0x98111019	RTERR_VMXCONTROLSMISSING	Fehlende Funktion in Intel VT-x Erweiterung.
0x101A	4122	0x9811101A	RTERR_VMXENABLEFAILS	Aktivieren von Intel VT-x schlägt fehl.

Spezifische positive HRESULT Return Codes:

HRESULT	Name	Beschreibung
0x0000_0000	S_OK	Kein Fehler.
0x0000_0001	S_FALSE	Kein Fehler. Bsp.: erfolgreiche Abarbeitung, bei der jedoch ein negatives oder unvollständiges Ergebnis erzielt wurde.
0x0000_0203	S_PENDING	Kein Fehler. Bsp.: erfolgreiche Abarbeitung, bei der jedoch noch kein Ergebnis vorliegt.
0x0000_0256	S_WATCHDOG_TIMEOUT	Kein Fehler. Bsp.: erfolgreiche Abarbeitung, bei der jedoch eine Zeitüberschreitung eintrat.

TCP Winsock-Fehlercodes

Hex	Dec	Name	Beschreibung
0x274C	10060	WSAETIMEDOUT	Verbindungs Timeout aufgetreten - Fehler beim Herstellen der Verbindung, da die Gegenstelle nach einer bestimmten Zeitspanne nicht ordnungsgemäß reagiert hat, oder die hergestellte Verbindung konnte nicht aufrecht erhalten werden, da der verbundene Host nicht reagiert hat.
0x274D	10061	WSAECONNREFUSED	Verbindung abgelehnt - Es konnte keine Verbindung hergestellt werden, da der Zielcomputer dies explizit abgelehnt hat. Dieser Fehler resultiert normalerweise aus dem Versuch, eine Verbindung mit einem Dienst herzustellen, der auf dem fremden Host inaktiv ist—das heißt, einem Dienst, für den keine Serveranwendung ausgeführt wird.
0x2751	10065	WSAEHOSTUNREACH	Keine Route zum Host - Ein Socketvorgang bezog sich auf einen nicht verfügbaren Host.
Weitere Winsock-Fehlercodes: Win32-Fehlercodes			

6.3 Interne Fehlercodes des TC3 XML Servers

Code	Beschreibung	Symbolischer Name
0x00008000	Interner Fehler	XMLSRVERROR_INTERNAL
0x00008001	Handle nicht gefunden.	XMLSRVERROR_NOTFOUND
0x00008002	Fehler beim Parsen der Xml-Datei	XMLSRVERROR_PARSERERROR
0x00008003	Inkompatibler Datentyp	XMLSRVERROR_INCOMPATIBLE
0x00008004	Fehler beim Allozieren von Speicher	XMLSRVERROR_NOMEMORY
0x00008005	Fehler beim Hinzufügen eines XML-Knotens	XMLSRVERROR_ADDNODE
0x00008006	Ungültiger sXPath	XMLSRVERROR_INVALIDXPATH
0x00008007	Ungültiger String in TC	XMLSRVERROR_INVALIDSTRING
0x00008800	Ungültiges Handle des Clients	XMLSRVERROR_INVALIDCLIENTHANDLE
0x0008900	Verwendung der falschen TcXmlDataSrv.exe (für TC2 statt TC3)	XMLSRVERROR_INVALIDTWINCATVERSION

6.4 FAQ - Häufig gestellte Fragen und Antworten

In diesem Bereich werden häufig gestellte Fragen beantwortet, um Ihnen die Arbeit mit XML Server zu erleichtern.
Wenn Sie noch weitere Fragen haben, kontaktieren Sie bitte unseren Support +49(0)5246/963-157)

Welche Informationen werden in der XML-Datei gespeichert? [▶ 39]

Können mehrere Variablen gleichzeitig in eine XML-Datei geschrieben werden? [▶ 39]

Ist es möglich, auf eine XML-Datei im Netzwerk zuzugreifen? [▶ 39]

Kann der XML Server automatisch XML-Dateien erzeugen? [▶ 39]

Was passiert, wenn der XML Server eine Variable nicht konvertieren kann (z.B. wenn eine Integer-Variable in der XML-Datei den Wert „hallo“ hat)? [▶ 39]

Werden Alias-Datentypen unterstützt? [▶ 39]

? Welche Informationen werden in der XML-Datei gespeichert?
! In der XML-Datei wird lediglich der Name der Variable und ihr Wert gespeichert – nicht ihr Datentyp. Außerdem kann ab Produktversion 3.2.31.0 ein SPS-Kommentar mit in die XML Datei geschrieben werden. Weitere Informationen dazu finden Sie im Abschnitt „Getting Started [▶ 24]“.

? Können mehrere Variablen gleichzeitig in eine XML-Datei geschrieben werden?
! Ja, dazu müssen die Variablen in TwinCAT in einer Struktur zusammengefasst werden. Geschrieben wird dann eine Variable vom Typ dieser Struktur. Diese Möglichkeit wird in den Beispielen [▶ 24] gezeigt.

? Ist es möglich, auf eine XML-Datei im Netzwerk zuzugreifen?
! Nein, diese Möglichkeit besteht nicht. sPath kann nur auf das lokale Filesystem des Rechners zeigen.

? Kann der XML Server automatisch XML-Dateien erzeugen?
! Ja, dies ist abhängig vom gewählten Modus. Um XML-Dateien automatisch zu erzeugen und fehlende Inhalte zu ergänzen, setzen Sie nMode:=1 (XMLSRV_ADDMISSING). Ordner werden nicht automatisch erzeugt!

? Was passiert, wenn der XML Server eine Variable nicht konvertieren kann (z.B. wenn eine Integer-Variable in der XML-Datei den Wert „hallo“ hat)?
! In diesem Fall wird die Variable übersprungen. Es kommt nicht zu einem Fehler.

? Werden Alias-Datentypen unterstützt?

! Ja, Alias-Datentypen werden ab der Produktversion 3.2.31.0 unterstützt.

Mehr Informationen:
www.beckhoff.de/tf6421

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Deutschland
Telefon: +49 5246 9630
info@beckhoff.de
www.beckhoff.de