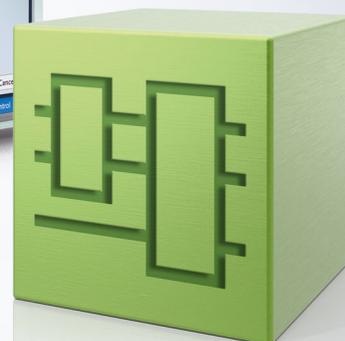
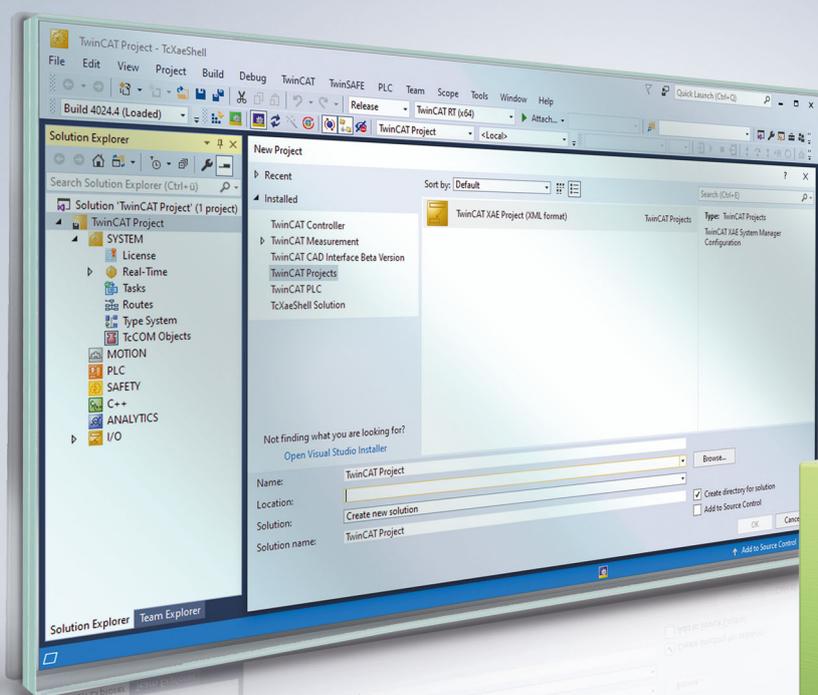


Handbuch | DE

## TE1000

TwinCAT 3 | PLC-Bibliothek: Tc3\_EventLogger





# Inhaltsverzeichnis

<b>1</b>	<b>Vorwort</b> .....	<b>7</b>
1.1	Hinweise zur Dokumentation .....	7
1.2	Zu Ihrer Sicherheit.....	8
1.3	Hinweise zur Informationssicherheit .....	9
<b>2</b>	<b>Übersicht</b> .....	<b>10</b>
<b>3</b>	<b>Funktionen und Funktionsbausteine</b> .....	<b>11</b>
3.1	Asynchrone Textanfragen .....	11
3.1.1	FB_AsyncStrResult .....	11
3.1.2	FB_RequestCauseRemedy .....	12
3.1.3	FB_RequestEventClassDetails .....	15
3.1.4	FB_RequestEventClassName .....	18
3.1.5	FB_RequestEventDetails .....	20
3.1.6	FB_RequestEventText .....	24
3.1.7	FB_RequestTranslation .....	27
3.1.8	FB_TcCauseRemedy .....	30
3.1.9	FB_TcDetail .....	32
3.1.10	F_GetEventClassName .....	34
3.1.11	F_GetEventText .....	35
3.2	EventEntry-Konvertierung .....	36
3.2.1	AdsErr_TO_TcEventEntry.....	36
3.2.2	HRESULTAdsErr_TO_TcEventEntry.....	37
3.2.3	TcEventEntry_TO_AdsErr.....	37
3.2.4	TcEventEntry_TO_HRESULTAdsErr.....	38
3.3	Filter .....	39
3.3.1	FB_TcClearLoggedEventsSettings .....	39
3.3.2	FB_TcEventCsvExportSettings.....	41
3.3.3	FB_TcEventFilter .....	41
3.4	RemoteEventLogger .....	44
3.4.1	FB_RemoteListenerBase .....	45
3.4.2	FB_TcRemoteEventLogger.....	50
3.5	FB_ListenerBase2.....	57
3.5.1	Execute .....	58
3.5.2	OnAlarmCleared .....	58
3.5.3	OnAlarmConfirmed .....	59
3.5.4	OnAlarmDisposed .....	59
3.5.5	OnAlarmRaised.....	60
3.5.6	OnMessageSent .....	60
3.5.7	Subscribe .....	60
3.5.8	Subscribe2 .....	61
3.5.9	Unsubscribe .....	61
3.6	FB_TcAlarm .....	62
3.6.1	Clear.....	64
3.6.2	Confirm.....	65
3.6.3	Create .....	66

3.6.4	CreateEx .....	66
3.6.5	Raise .....	67
3.6.6	SetJsonAttribute .....	68
3.7	FB_TcArguments .....	68
3.7.1	IsEmpty .....	70
3.8	FB_TcEvent .....	70
3.9	FB_TcEventBase .....	72
3.9.1	EqualsTo .....	73
3.9.2	EqualsToEventClass .....	74
3.9.3	EqualsToEventEntry .....	74
3.9.4	EqualsToEventEntryEx .....	75
3.9.5	GetJsonAttribute .....	75
3.9.6	Release .....	76
3.9.7	RequestEventClassName .....	76
3.9.8	RequestEventText .....	77
3.9.9	ipArguments .....	78
3.9.10	ipSourceInfo .....	78
3.10	FB_TcEventLogger .....	78
3.10.1	ClearAlarms .....	79
3.10.2	ClearAllAlarms .....	79
3.10.3	ClearLoggedEvents .....	80
3.10.4	ConfirmAlarms .....	81
3.10.5	ConfirmAllAlarms .....	81
3.10.6	ExportLoggedEvents .....	82
3.10.7	GetAlarm .....	83
3.10.8	GetAlarmEx .....	83
3.10.9	IsAlarmRaised .....	84
3.10.10	IsAlarmRaisedEx .....	84
3.10.11	SendMessage .....	85
3.10.12	SendMessage2 .....	86
3.10.13	SendMessageEx .....	87
3.10.14	SendMessageEx2 .....	88
3.11	FB_TcMessage .....	89
3.11.1	Create .....	91
3.11.2	CreateEx .....	92
3.11.3	SetJsonAttribute .....	92
3.12	FB_TcSourceInfo .....	93
3.12.1	Clear .....	94
3.12.2	ExtendName .....	94
3.12.3	ResetToDefault .....	95
<b>4</b>	<b>Schnittstellen .....</b>	<b>96</b>
4.1	I_TcArguments .....	96
4.1.1	AddBlob .....	96
4.1.2	AddBool .....	97
4.1.3	AddByte .....	97
4.1.4	AddDint .....	98

4.1.5	AddDWord.....	98
4.1.6	AddEventReferenceId .....	99
4.1.7	AddEventReferenceIdGuid .....	99
4.1.8	AddInt.....	100
4.1.9	AddLInt.....	100
4.1.10	AddLReal .....	100
4.1.11	AddReal .....	101
4.1.12	AddSInt .....	101
4.1.13	AddString .....	102
4.1.14	AddUDint.....	102
4.1.15	AddUInt .....	103
4.1.16	AddULInt .....	103
4.1.17	AddUSInt.....	104
4.1.18	AddWord .....	104
4.1.19	AddWString.....	104
4.1.20	Clear.....	105
4.2	I_TcEventBase .....	105
4.2.1	EqualsTo.....	106
4.2.2	EqualsToEventClass.....	106
4.2.3	EqualsToEventEntry .....	107
4.2.4	EqualsToEventEntryEx .....	107
4.2.5	GetJsonAttribute .....	108
4.2.6	RequestEventClassName .....	108
4.2.7	RequestEventText.....	109
4.3	I_TcMessage.....	110
4.3.1	Send.....	110
4.4	I_TcSourceInfo.....	111
4.4.1	EqualsTo.....	111
<b>5</b>	<b>Datentypen.....</b>	<b>113</b>
5.1	TcEventEntry.....	113
5.2	TcEventSeverity .....	113
5.3	TcEventConfirmationState .....	113
<b>6</b>	<b>Globale Listen .....</b>	<b>115</b>
6.1	Global_Constants.....	115
6.2	GVL .....	115
6.3	Parameterlist .....	115
6.4	Global_Version.....	115
<b>7</b>	<b>Beispiele .....</b>	<b>116</b>
7.1	Tutorial .....	116
7.2	Beispiel ResultMessage.....	118
7.3	Beispiel Listener.....	118
7.4	Beispiel Filter.....	119



# 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, stets die aktuell gültige 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 weiterentwickelt.

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®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® und XTS® 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 Kennzeichnungen führen.



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

### Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.

Weitergabe sowie Vervielfältigung dieses Dokuments, Verwertung und Mitteilung seines Inhalts sind verboten, soweit nicht ausdrücklich gestattet.

Zuwendungen verpflichten zu Schadenersatz. Alle Rechte für den Fall der Patent-, Gebrauchsmuster- oder Geschmacksmustereintragung vorbehalten.

### Fremdmarken

In dieser Dokumentation können Marken Dritter verwendet werden. Die zugehörigen Markenvermerke finden Sie unter: <https://www.beckhoff.com/trademarks>.

## 1.2 Zu Ihrer Sicherheit

### Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.  
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

### 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.

### Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

### Warnungen vor Personenschäden

#### **GEFAHR**

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

#### **WARNUNG**

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

#### **VORSICHT**

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

### Warnung vor Umwelt- oder Sachschäden

#### **HINWEIS**

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

### Information zum Umgang mit dem Produkt



Diese Information beinhaltet z. B.:  
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.

## 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

Die SPS-Bibliothek Tc3\_EventLogger umfasst Funktionen und Funktionsbausteine zur Verwendung des TwinCAT 3 EventLogger.

Siehe auch: [Dokumentation TwinCAT 3 EventLogger](#)

### Funktionsbausteine zur Verwendung des TC3 EventLogger

<a href="#">FB_ListenerBase2</a> [▶ 57]	Basisimplementierung eines Ereignisbeobachters.
<a href="#">FB_TcAlarm</a> [▶ 62]	Repräsentiert einen Alarm vom TwinCAT 3 EventLogger.
<a href="#">FB_TcArguments</a> [▶ 68]	Definiert Argumente eines Ereignisses.
<a href="#">FB_TcEventLogger</a> [▶ 78]	Stellt den TwinCAT 3 EventLogger selbst dar.
<a href="#">FB_TcMessage</a> [▶ 89]	Repräsentiert eine Nachricht vom TwinCAT 3 EventLogger.
<a href="#">FB_TcSourceInfo</a> [▶ 93]	Definiert Quellinformation eines Ereignisses.

### Asynchrone Textanfragen

<a href="#">F_GetEventClassName</a> [▶ 34]	Triggert die asynchrone Anfrage des Namens einer Ereignisklasse.
<a href="#">F_GetEventText</a> [▶ 35]	Triggert die asynchrone Anfrage eines Ereignistextes.
<a href="#">FB_AsyncStrResult</a> [▶ 11]	Ermöglicht die asynchrone Anfrage eines Textes.
<a href="#">FB_RequestEventClassName</a> [▶ 18]	Ermöglicht die asynchrone Anfrage des Namens einer Ereignisklasse.
<a href="#">FB_RequestEventText</a> [▶ 24]	Ermöglicht die asynchrone Anfrage eines Ereignistextes in gewünschter Sprache.

### EventEntry-Konvertierung

<a href="#">AdsErr_TO_TcEventEntry</a> [▶ 36]	Konvertiert einen Standard-ADS-Fehler in ein TcEventEntry.
<a href="#">HRESULTAdsErr_TO_TcEventEntry</a> [▶ 37]	Konvertiert einen Standard-ADS-Fehler (HRESULT) in ein TcEventEntry.
<a href="#">TcEventEntry_TO_AdsErr</a> [▶ 37]	Konvertiert ein TcEventEntry in einen Standard-ADS-Fehler.
<a href="#">TcEventEntry_TO_hresultAdsErr</a> [▶ 38]	Konvertiert ein TcEventEntry in einen Standard-ADS-Fehler (HRESULT).

### 3 Funktionen und Funktionsbausteine

#### 3.1 Asynchrone Textanfragen

##### 3.1.1 FB\_AsyncStrResult



Dieser Funktionsbaustein ermöglicht die asynchrone Anfrage eines Textes.

**Methoden**

Name	Beschreibung
GetString [▶ 11]	Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode der angefragte Text abgeholt werden.

**Eigenschaften**

Name	Typ	Zugriff	Beschreibung
bBusy	BOOL	Get	TRUE, solange die Abarbeitung noch nicht abgeschlossen ist.
bError	BOOL	Get	TRUE, sobald ein Fehler eintritt.
hrErrorCode	HRESULT	Get	Gibt die Fehlerinformation aus, wenn bError TRUE ist.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

##### 3.1.1.1 GetString



Sobald bBusy = FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode der angefragte Text abgeholt werden.

**Syntax**

```
METHOD GetString : BOOL
VAR_INPUT
    sResult : REFERENCE TO STRING;
    nResult : UDINT;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Puffervariable für den angefragten Text
nResult	UDINT	Puffergröße in Bytes

 **Rückgabewert**

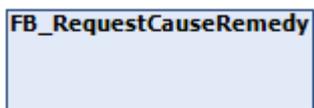
Name	Typ	Beschreibung
GetString	BOOL	Liefert TRUE, wenn der Text zugewiesen werden konnte. Liefert FALSE, wenn der Text nicht vollständig zugewiesen werden konnte, weil die angegebene Puffervariable zu klein ist.

**Beispiel**

Die Methode darf erst dann aufgerufen werden, wenn mittels bBusy = FALSE und bError = FALSE signalisiert wurde, dass ein Text zur Verfügung steht.

```
IF NOT fb.bBusy AND NOT fb.bError THEN
    bGetStringSuccess := fb.GetString(sText, sizeof(sText));
END_IF
```

### 3.1.2 FB\_RequestCauseRemedy



Dieser Funktionsbaustein ermöglicht die asynchrone Anfrage von Ursache/Behebung in gewünschter Sprache zu einem Ereignis.

 **Methoden**

Name	Beschreibung
<a href="#">Clear</a> [ <a href="#">▶ 12</a> ]	Diese Methode löscht das zuletzt abgefragte Ergebnis.
<a href="#">Get</a> [ <a href="#">▶ 13</a> ]	Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode ein Ergebnis aus der abgefragten Liste abgeholt werden.
<a href="#">Request</a> [ <a href="#">▶ 13</a> ]	Der Aufruf dieser Methode triggert die asynchrone Anfrage.
<a href="#">RequestRemote</a> [ <a href="#">▶ 14</a> ]	Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
bBusy	BOOL	GET	TRUE, solange die Abarbeitung noch nicht abgeschlossen ist.
bError	BOOL	GET	TRUE, sobald ein Fehler eintritt.
hrErrorCode	HRESULT	GET	Gibt die Fehlerinformation aus, wenn bError TRUE ist.
nCount	UDINT	GET	Anzahl der gefundenen Ursachen/behobenen Elemente

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4026.0	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>=3.3.7.0)

#### 3.1.2.1 Clear



Diese Methode löscht das zuletzt abgefragte Ergebnis.

**Syntax**

METHOD Clear

**3.1.2.2 Get**



Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode ein Ergebnis aus der abgefragten Liste abgeholt werden.

**Syntax**

```
METHOD Get : HRESULT
VAR_INPUT
  nIndex : UDINT;
  fbCauseRemedy : REFERENCE TO FB_TcCauseRemedy;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
nIndex	UDINT	Index in der Liste von Ursache/Behebung-Elementen, der abzufragen ist.
fbCauseRemedy	Reference to FB_TcCauseRemedy [►_30]	Referenz auf einen FB_TcCauseRemedy, in dem die Texte abgelegt werden.

**Ausgänge**

Name	Typ	Beschreibung
Get	HRESULT	S_OK bei Erfolg ADS_E_INVALIDSTATE, wenn die Abfrage noch läuft. ADS_E_INVALIDSIZE, wenn ein Index abgefragt wird, der nicht belegt ist.

**3.1.2.3 Request**



Der Aufruf dieser Methode triggert die asynchrone Anfrage.

**Syntax**

```
METHOD Request : HRESULT
VAR_INPUT
  eventClass : GUID;
  nEventId : UDINT;
  nLangId : DINT; // English(US)=1033 ; German(Germay)=1031
END_VAR
```

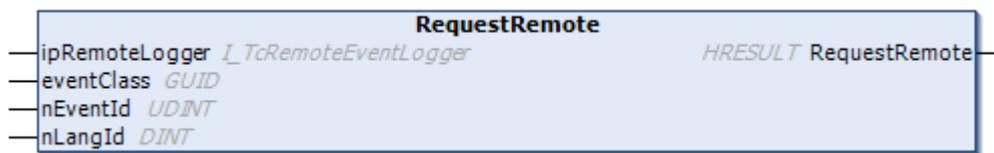
 **Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	Spezifiziert die Ereignisklasse.
nEventId	UDINT	ID von dem Ereignis
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.

 **Ausgänge**

Name	Typ	Beschreibung
Request	HRESULT	Liefert mögliche Fehlerinformationen. S_OK, wenn die Abfrage erfolgreich ist. ADS_E_INVALIDSTATE, wenn ein illegaler Zustand des Zielsystems vorliegt. ADS_E_NOMEMORY, wenn kein Speicher bereitsteht, um das Ergebnis abzulegen. ADS_E_PENDING, wenn die Abfrage noch bearbeitet wird.

**3.1.2.4 RequestRemote**



Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

**Syntax**

```
METHOD RequestRemote : HRESULT
VAR_INPUT
    ipRemoteLogger : I_TcRemoteEventLogger;
    eventClass     : GUID;
    nEventId       : UDINT;
    nLangId        : DINT; // English(US)=1033 ; German(Germay)=1031
END_VAR
```

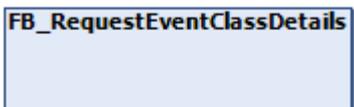
 **Eingänge**

Name	Typ	Beschreibung
ipRemoteLogger	I_TcRemoteEventLogger	Instanz vom I_TcRemoteEventLogger, der das Zielsystem beschreibt.
eventClass	GUID	Spezifiziert die Ereignisklasse.
nEventId	UDINT	ID von dem Ereignis
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.

 **Ausgänge**

Name	Typ	Beschreibung
RequestRemote	HRESULT	Liefert mögliche Fehlerinformationen. S_OK, wenn die Abfrage erfolgreich ist. ADS_E_INVALIDSTATE, wenn ein illegaler Zustand des Zielsystems vorliegt. ADS_E_NOMEMORY, wenn kein Speicher bereitsteht, um das Ergebnis abzulegen. ADS_E_PENDING, wenn die Abfrage noch bearbeitet wird.

### 3.1.3 FB\_RequestEventClassDetails



Mit diesem Funktionsbaustein können die Details einer Ereignisklasse in Form von [FB\\_TcDetail \[▶ 32\]](#) abgefragt werden.

Jedes „Detail“ (DescriptionText, Descriptionurl, Comment) – vgl. beim Anlegen des Events) hat dabei einen Index und zu jedem Index wird der Key ((DescriptionText, DescriptionUrl, Comment) sowie der Wert geliefert.

 **Methoden**

Name	Beschreibung
<a href="#">Clear [▶ 15]</a>	Diese Methode löscht das zuletzt abgefragte Ergebnis.
<a href="#">Get [▶ 16]</a>	Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode ein Ergebnis aus der abgefragten Liste abgeholt werden.
<a href="#">Request [▶ 16]</a>	Der Aufruf dieser Methode triggert die asynchrone Anfrage.
<a href="#">RequestRemote [▶ 17]</a>	Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
bBusy	BOOL	GET	TRUE, solange die Abarbeitung noch nicht abgeschlossen ist.
bError	BOOL	GET	TRUE, sobald ein Fehler eintritt.
hrErrorCode	HRESULT	GET	Gibt die Fehlerinformation aus, wenn bError TRUE ist.
nCount	UDINT	GET	Anzahl der gefundenen Ursachen/behobenen Elemente

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4026.0	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>=3.3.7.0)

#### 3.1.3.1 Clear

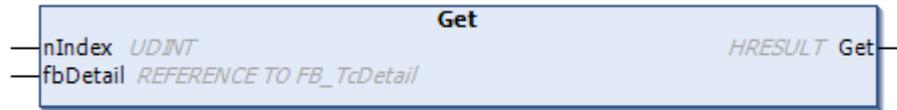


Diese Methode löscht das zuletzt abgefragte Ergebnis.

**Syntax**

```
METHOD Clear
```

**3.1.3.2 Get**



Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode ein Ergebnis aus der abgefragten Liste abgeholt werden.

**Syntax**

```
METHOD Get      : HRESULT
VAR_INPUT
  nIndex      : UDINT;
  fbDetail    : REFERENCE TO FB_TcDetail;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
nIndex	UDINT	Index in der Liste von Ursache/Behebung-Elementen, der abzufragen ist.
fbDetail	Reference to <a href="#">FB_TcDetail</a> <a href="#">▶ 32</a>	Referenz auf einen FB_TcDetail, in dem die Details abgelegt werden.

**Ausgänge**

Name	Typ	Beschreibung
Get	HRESULT	Liefert S_OK bei Erfolg. ADS_E_INVALIDSTATE, wenn die Abfrage noch läuft. ADS_E_INVALIDSIZE, wenn ein Index abgefragt wird, der nicht belegt ist.

**3.1.3.3 Request**



Der Aufruf dieser Methode triggert die asynchrone Anfrage.

**Syntax**

```
METHOD Request : HRESULT
VAR_INPUT
  eventClass : GUID;
  nLangId    : DINT; // English(US)=1033 ; German(Germay)=1031
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	Spezifiziert die Ereignisklasse.
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.

 **Ausgänge**

Name	Typ	Beschreibung
Request	HRESULT	Liefert mögliche Fehlerinformationen. S_OK, wenn die Abfrage erfolgreich ist. ADS_E_INVALIDSTATE, wenn ein illegaler Zustand des Zielsystems vorliegt. ADS_E_NOMEMORY, wenn kein Speicher bereitsteht, um das Ergebnis abzulegen. ADS_E_PENDING, wenn die Abfrage noch bearbeitet wird.

**3.1.3.4 RequestRemote**



Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

**Syntax**

```
METHOD RequestRemote : HRESULT
VAR_INPUT
    ipRemoteLogger : I_TcRemoteEventLogger;
    eventClass      : GUID;
    nLangId         : DINT; // English(US)=1033 ; German(Germay)=1031
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
ipRemoteLogger	I_TcRemoteEventLogger	Instanz vom I_TcRemoteEventLogger, der das Zielsystem beschreibt.
eventClass	GUID	Spezifiziert die Ereignisklasse.
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.

## Ausgänge

Name	Typ	Beschreibung
RequestRemote	HRESULT	Liefert mögliche Fehlerinformationen. S_OK, wenn die Abfrage erfolgreich ist. ADS_E_INVALIDSTATE, wenn ein illegaler Zustand des Zielsystems vorliegt. ADS_E_NOMEMORY, wenn kein Speicher bereitsteht, um das Ergebnis abzulegen. ADS_E_PENDING, wenn die Abfrage noch bearbeitet wird.

### 3.1.4 FB\_RequestEventClassName

#### FB\_RequestEventClassName

Dieser Funktionsbaustein ermöglicht die asynchrone Anfrage des Namens einer Ereignisklasse.

## Methoden

Name	Beschreibung
Clear [ <a href="#">▶ 18</a> ]	Diese Methode löscht das zuletzt abgefragte Ergebnis.
GetString [ <a href="#">▶ 19</a> ]	Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode der angefragte Text abgeholt werden.
Request [ <a href="#">▶ 19</a> ]	Der Aufruf dieser Methode triggert die asynchrone Textanfrage.
RequestRemote [ <a href="#">▶ 20</a> ]	Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

## Eigenschaften

Name	Typ	Zugriff	Beschreibung
bBusy	BOOL	Get	TRUE, solange die Abarbeitung noch nicht abgeschlossen ist.
bError	BOOL	Get	TRUE, sobald ein Fehler eintritt.
hrErrorCode	HRESULT	Get	Gibt die Fehlerinformation aus, wenn bError TRUE ist.

## Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 3.1.4.1 Clear

#### Clear

Diese Methode löscht das zuletzt abgefragte Ergebnis.

## Syntax

```
METHOD Clear
```

### 3.1.4.2 GetString



Sobald bBusy = FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode der angefragte Text abgeholt werden.

#### Syntax

```
METHOD GetString : BOOL
VAR_INPUT
    sResult : REFERENCE TO STRING;
    nResult : UDINT;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Puffervariable für den angefragten Text
nResult	UDINT	Puffergröße in Bytes

#### Rückgabewert

Name	Typ	Beschreibung
GetString	BOOL	Liefert TRUE, wenn der Text zugewiesen werden konnte. Liefert FALSE, wenn der Text nicht vollständig zugewiesen werden konnte, weil die angegebene Puffervariable zu klein ist.

#### Beispiel

Die Methode darf erst dann aufgerufen werden, wenn mittels bBusy = FALSE und bError = FALSE signalisiert wurde, dass ein Text zur Verfügung steht.

```
IF NOT fb.bBusy AND NOT fb.bError THEN
    bGetStringSuccess := fb.GetString(sText, sizeof(sText));
END_IF
```

### 3.1.4.3 Request



Der Aufruf dieser Methode triggert die asynchrone Textanfrage.

#### Syntax

```
METHOD Request : HRESULT
VAR_INPUT
    eventClass : GUID;
    nLangId : DINT;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse.
nLangId	DINT	Spezifiziert die Sprach-ID Englisch (en-US) = 1033 Deutsch (de-DE) = 1031

 **Rückgabewert**

Name	Typ	Beschreibung
Request	HRESULT	Liefert mögliche Fehlerinformationen.

### 3.1.4.4 RequestRemote



Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

**Syntax**

```
METHOD RequestRemote : HRESULT
VAR_INPUT
    ipRemoteLogger : I_TcRemoteEventLogger;
    eventClass     : GUID;
    nLangId        : DINT; // English(US)=1033 ; German(Germay)=1031
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
ipRemoteLogger	I_TcRemoteEventLogger	Instanz vom I_TcRemoteEventLogger, der das Zielsystem beschreibt.
eventClass	GUID	Spezifiziert die Ereignisklasse.
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.

 **Ausgänge**

Name	Typ	Beschreibung
RequestRemote	HRESULT	Liefert mögliche Fehlerinformationen. S_OK, wenn die Abfrage erfolgreich ist. ADS_E_INVALIDSTATE, wenn ein illegaler Zustand des Zielsystems vorliegt. ADS_E_NOMEMORY, wenn kein Speicher bereitsteht, um das Ergebnis abzulegen. ADS_E_PENDING, wenn die Abfrage noch bearbeitet wird.

### 3.1.5 FB\_RequestEventDetails



Mit diesem Funktionsbaustein können die Details eines Ereignisses in Form von [FB\\_TcDetail](#) [► 32] abgefragt werden.

 **Methoden**

Name	Beschreibung
<a href="#">Clear</a> [ <a href="#">▶ 21</a> ]	Diese Methode löscht das zuletzt abgefragte Ergebnis.
<a href="#">Get</a> [ <a href="#">▶ 21</a> ]	Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode ein Ergebnis aus der abgefragten Liste abgeholt werden.
<a href="#">Request</a> [ <a href="#">▶ 22</a> ]	Der Aufruf dieser Methode triggert die asynchrone Anfrage.
<a href="#">RequestRemote</a> [ <a href="#">▶ 23</a> ]	Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
bBusy	BOOL	GET	TRUE, solange die Abarbeitung noch nicht abgeschlossen ist.
bError	BOOL	GET	TRUE, sobald ein Fehler eintritt.
hrErrorCode	HRESULT	GET	Gibt die Fehlerinformation aus, wenn bError TRUE ist.
nCount	UDINT	GET	Anzahl der gefundenen Ursachen/behobenen Elemente

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4026.0	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>=3.3.7.0)

**3.1.5.1 Clear**



Diese Methode löscht das zuletzt abgefragte Ergebnis.

**Syntax**

```
METHOD Clear
```

**3.1.5.2 Get**



Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode ein Ergebnis aus der abgefragten Liste abgeholt werden.

**Syntax**

```
METHOD Get : HRESULT
VAR_INPUT
  nIndex : UDINT;
  fbDetail : REFERENCE TO FB_TcDetail;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nIndex	UDINT	Index in der Liste von Ursachen/behobenen Elementen, der abzufragen ist.
fbDetail	Reference to <a href="#">FB_TcDetail</a> [▶_32]	Referenz auf einen FB_TcDetail, in dem die Details abgelegt werden.

 **Ausgänge**

Name	Typ	Beschreibung
Get	HRESULT	S_OK bei Erfolg ADS_E_INVALIDSTATE, wenn die Abfrage noch läuft. ADS_E_INVALIDSIZE, wenn ein Index abgefragt wird, der nicht belegt ist.

**3.1.5.3 Request**



Der Aufruf dieser Methode triggert die asynchrone Anfrage.

**Syntax**

```

METHOD Request : HRESULT
VAR_INPUT
    eventClass : GUID;
    nEventId : UDINT;
    nLangId : DINT; // English(US)=1033 ; German(Germay)=1031
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	Spezifiziert die Ereignisklasse.
nEventId	UDINT	ID von dem Ereignis
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.

 **Ausgänge**

Name	Typ	Beschreibung
Request	HRESULT	Liefert mögliche Fehlerinformationen. S_OK, wenn die Abfrage erfolgreich ist. ADS_E_INVALIDSTATE, wenn ein illegaler Zustand des Zielsystems vorliegt. ADS_E_NOMEMORY, wenn kein Speicher bereitsteht, um das Ergebnis abzulegen. ADS_E_PENDING, wenn die Abfrage noch bearbeitet wird.

**3.1.5.4 RequestRemote**



Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

**Syntax**

```

METHOD RequestRemote : HRESULT
VAR_INPUT
    ipRemoteLogger : I_TcRemoteEventLogger;
    eventClass      : GUID;
    nEventId        : UDINT;
    nLangId         : DINT; // English(US)=1033 ; German(Germay)=1031
END_VAR
  
```

 **Eingänge**

Name	Typ	Beschreibung
ipRemoteLogger	I_TcRemoteEventLogger	Instanz vom I_TcRemoteEventLogger, der das Zielsystem beschreibt.
eventClass	GUID	Spezifiziert die Ereignisklasse.
nEventId	UDINT	ID von dem Ereignis
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.

 **Ausgänge**

Name	Typ	Beschreibung
RequestRemote	HRESULT	Liefert mögliche Fehlerinformationen. S_OK, wenn die Abfrage erfolgreich ist. ADS_E_INVALIDSTATE, wenn ein illegaler Zustand des Zielsystems vorliegt. ADS_E_NOMEMORY, wenn kein Speicher bereitsteht, um das Ergebnis abzulegen. ADS_E_PENDING, wenn die Abfrage noch bearbeitet wird.

### 3.1.6 FB\_RequestEventText



Dieser Funktionsbaustein ermöglicht die asynchrone Anfrage eines Ereignistextes in gewünschter Sprache.

#### Methoden

Name	Beschreibung
Clear [▶ 26]	Diese Methode löscht das zuletzt abgefragte Ergebnis.
GetString [▶ 24]	Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode der angefragte Text abgeholt werden.
Request [▶ 25]	Der Aufruf dieser Methode triggert die asynchrone Textanfrage.
RequestRemote [▶ 26]	Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

#### Eigenschaften

Name	Typ	Zugriff	Beschreibung
bBusy	BOOL	Get	TRUE, solange die Abarbeitung noch nicht abgeschlossen ist.
bError	BOOL	Get	TRUE, sobald ein Fehler eintritt.
hrErrorCode	HRESULT	Get	Gibt die Fehlerinformation aus, wenn bError TRUE ist.

#### Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

#### 3.1.6.1 GetString



Sobald bBusy = FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode der angefragte Text abgeholt werden.

#### Syntax

```
METHOD GetString : BOOL
VAR_INPUT
    sResult : REFERENCE TO STRING;
    nResult : UDINT;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Puffervariable für den angefragten Text
nResult	UDINT	Puffergröße in Bytes

 **Rückgabewert**

Name	Typ	Beschreibung
GetString	BOOL	Liefert TRUE, wenn der Text zugewiesen werden konnte. Liefert FALSE, wenn der Text nicht vollständig zugewiesen werden konnte, weil die angegebene Puffervariable zu klein ist.

**Beispiel**

Die Methode darf erst dann aufgerufen werden, wenn mittels bBusy = FALSE und bError = FALSE signalisiert wurde, dass ein Text zur Verfügung steht.

```
IF NOT fb.bBusy AND NOT fb.bError THEN
    bGetStringSuccess := fb.GetString(sText, SIZEOF(sText));
END_IF
```

**3.1.6.2 Request**



Der Aufruf dieser Methode triggert die asynchrone Textanfrage.

**Syntax**

```
METHOD Request : BOOL
VAR_INPUT
    eventClass : GUID;
    nEventId : UDINT;
    nLangId : DINT;
    ipArgs : I_TcArguments;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	Spezifiziert die Ereignisklasse.
nEventId	UDINT	ID von dem Ereignis.
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.
ipArgs	I_TcArguments <a href="#">[► 96]</a>	Optionale Angabe von Argumenten.

 **Rückgabewert**

Name	Typ	Beschreibung
Request	HRESULT	Liefert mögliche Fehlerinformationen.

### 3.1.6.3 RequestRemote



Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

#### Syntax

```
METHOD RequestRemote : HRESULT
VAR_INPUT
    ipRemoteLogger : I_TcRemoteEventLogger;
    eventClass     : GUID;
    nEventId       : UDINT;
    nLangId        : DINT;
    ipArgs         : I_TcArguments;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
ipRemoteLogger	I_TcRemoteEventLogger	Instanz vom I_TcRemoteEventLogger, der das Zielsystem beschreibt.
eventClass	GUID	Spezifiziert die Ereignisklasse.
nEventId	UDINT	ID von dem Ereignis.
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.
ipArgs	I_TcArguments	Optionale Angabe von Argumenten.

#### Rückgabewert

Name	Typ	Beschreibung
RequestRemote	HRESULT	

#### Sehen Sie dazu auch

[I\\_TcArguments](#) [► 96]

### 3.1.6.4 Clear



Diese Methode löscht das zuletzt abgefragte Ergebnis.

#### Syntax

```
METHOD Clear
```

### 3.1.7 FB\_RequestTranslation

**FB\_RequestTranslation**

Dieser Funktionsbaustein liefert zu einem Text und einer gewünschten Sprache eine Übersetzung.

Die Übersetzungen werden durch eine Ereignisklasse referenziert, welche bei der Abfrage angegeben wird. Sie wird an dieser Stelle also nicht als Eventklasse verwendet, sondern als Übersetzungstabelle.

 **Methoden**

Name	Beschreibung
Clear [ <a href="#">▶ 29</a> ]	Diese Methode löscht das zuletzt abgefragte Ergebnis.
GetString [ <a href="#">▶ 27</a> ]	Sobald bBusy FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode der angefragte Text abgeholt werden.
Request [ <a href="#">▶ 28</a> ]	Der Aufruf dieser Methode triggert die asynchrone Textanfrage.
RequestRemote [ <a href="#">▶ 29</a> ]	Der Aufruf dieser Methode triggert die asynchrone Anfrage auf einem entfernten Zielsystem.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
bBusy	BOOL	Get	TRUE, solange die Abarbeitung noch nicht abgeschlossen ist.
bError	BOOL	Get	TRUE, sobald ein Fehler eintritt.
hrErrorCode	HRESULT	Get	Gibt die Fehlerinformation aus, wenn bError TRUE ist.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4026.0	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>=3.3.7.0)

#### 3.1.7.1 GetString



Sobald bBusy = FALSE ist und wenn kein Fehler aufgetreten ist (bError = FALSE), kann mittels dieser Methode der angefragte Text abgeholt werden.

**Syntax**

```
METHOD GetString : BOOL
VAR_INPUT
    sResult : REFERENCE TO STRING;
    nResult : UDINT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Puffervariable für den angefragten Text
nResult	UDINT	Puffergröße in Bytes

**📌 Rückgabewert**

Name	Typ	Beschreibung
GetString	BOOL	Liefert TRUE, wenn der Text zugewiesen werden konnte. Liefert FALSE, wenn der Text nicht vollständig zugewiesen werden konnte, weil die angegebene Puffervariable zu klein ist.

**Beispiel**

Die Methode darf erst dann aufgerufen werden, wenn mittels bBusy = FALSE und bError = FALSE signalisiert wurde, dass ein Text zur Verfügung steht.

```
IF NOT fb.bBusy AND NOT fb.bError THEN
    bGetStringSuccess := fb.GetString(sText, sizeof(sText));
END_IF
```

**3.1.7.2 Request**



Der Aufruf dieser Methode triggert die asynchrone Textanfrage.

**Syntax**

```
METHOD Request : BOOL
VAR_INPUT
    eventClass : GUID;
END_VAR
VAR_IN_OUT CONSTANT
    text : STRING;
END_VAR
VAR_INPUT
    nLangId : DINT;
    ipArgs : I_TcArguments;
END_VAR
```

**📌 Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	Spezifiziert die Ereignisklasse, welche als Basis für die Abfrage verwendet werden soll.
text	STRING	Der zu übersetzende Text
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.
ipArgs	I_TcArguments <a href="#"> ▶ 96</a>	Optionale Angabe von Argumenten.

**📌 Rückgabewert**

Name	Typ	Beschreibung
Request	HRESULT	Liefert mögliche Fehlerinformationen.

### 3.1.7.3 RequestRemote



Der Aufruf dieser Methode die asynchrone Anfrage auf einem entfernten Zielsystem.

#### Syntax

```
METHOD RequestRemote : HRESULT
VAR_INPUT
    ipRemoteLogger : I_TcRemoteEventLogger
    eventClass     : GUID;
END_VAR
VAR_IN_OUT CONSTANT
    text          : STRING;
END_VAR
VAR_INPUT
    nLangId       : DINT;
    ipArgs        : I_TcArguments;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
ipRemoteLogger	I_TcRemoteEventLogger	Instanz vom I_TcRemoteEventLogger, der das Zielsystem beschreibt.
eventClass	GUID	Spezifiziert die Ereignisklasse, welche als Basis für die Abfrage verwendet werden soll.
text	STRING	Der zu übersetzende Text
nLangId	DINT	Spezifiziert die Sprach-ID: Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ... Der Enum TcLcid kann verwendet werden.
ipArgs	I_TcArguments <a href="#"> &gt; 96</a>	Optionale Angabe von Argumenten.

#### Rückgabewert

Name	Typ	Beschreibung
RequestRemote	HRESULT	Liefert mögliche Fehlerinformationen. S_OK, wenn die Abfrage erfolgreich ist. ADS_E_INVALIDSTATE, wenn ein invalider Zustand des Zielsystems vorliegt. ADS_E_NOMEMORY, wenn kein Speicher bereitsteht, um das Ergebnis abzulegen. ADS_E_PENDING, wenn die Abfrage noch bearbeitet wird.

### 3.1.7.4 Clear

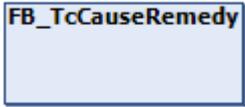


Diese Methode löscht das zuletzt abgefragte Ergebnis.

**Syntax**

METHOD Clear

### 3.1.8 FB\_TcCauseRemedy



Dieser Funktionsbaustein wird verwendet, um die Ursache/Behebung zu einem Alarm darzustellen. Dieser Funktionsbaustein wird mittels `FB_RequestCauseRemedy` [12] genutzt.

**Methoden**

Name	Beschreibung
<code>GetCause</code> [30]	Liefert die Ursache.
<code>GetId</code> [31]	Liefert den Index der Ursache/Behebung.
<code>GetRemedy</code> [31]	Liefert die Behebung.
<code>Release</code> [32]	Löscht den internen Speicher des Bausteins.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4026.0	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>=3.3.7.0)

#### 3.1.8.1 GetCause



Diese Methode liefert die Ursache.

**Syntax**

```
METHOD GetCause : BOOL
VAR_INPUT
    sResult      : REFERENCE TO STRING;
    nResult      : UDINT; // buffer size in bytes
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Referenz auf einen String, in dem das Ergebnis abgelegt werden soll.
nResult	UDINT	Länge des sResult

**Ausgänge**

Name	Typ	Beschreibung
GetCause	BOOL	Liefert den Erfolg, ob eine Ursache bereitgestellt wurde.

### 3.1.8.2 GetId



Diese Methode liefert den Index der Ursache/Behebung.

#### Syntax

```
METHOD GetId : BOOL
VAR_INPUT
    sResult : REFERENCE TO STRING;
    nResult : UDINT; // buffer size in bytes
END_VAR
```

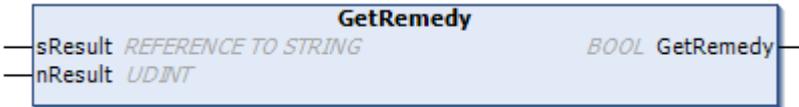
#### Eingänge

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Referenz auf einen String, in dem das Ergebnis abgelegt werden soll.
nResult	UDINT	Länge des sResult

#### Ausgänge

Name	Typ	Beschreibung
GetId	BOOL	Liefert den Erfolg, ob eine Id bereitgestellt wurde.

### 3.1.8.3 GetRemedy



Diese Methode liefert die Behebung.

#### Syntax

```
METHOD GetRemedy : BOOL
VAR_INPUT
    sResult : REFERENCE TO STRING;
    nResult : UDINT; // buffer size in bytes
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Referenz auf einen String, in dem das Ergebnis abgelegt werden soll.
nResult	UDINT	Länge des sResult

#### Ausgänge

Name	Typ	Beschreibung
GetRemedy	BOOL	Liefert den Erfolg, ob eine Behebung bereitgestellt wurde.

### 3.1.8.4 Release



Diese Methode löscht den internen Speicher des Bausteins.

#### Syntax

METHOD Release : HRESULT

#### Ausgänge

Name	Typ	Beschreibung
Release	HRESULT	Liefert den Erfolg, ob der Speicher aufgeräumt wurde.

### 3.1.9 FB\_TcDetail



Dieser Funktionsbaustein wird verwendet um die Details zu einem Ereignis darzustellen.

Dieser Funktionsbaustein wird mittels [FB\\_RequestEventClassDetails \[► 15\]](#) oder [FB\\_RequestEventDetails \[► 20\]](#) verwendet.

#### Methoden

Name	Beschreibung
<a href="#">GetComment [► 33]</a>	Liefert den Kommentar.
<a href="#">GetName [► 33]</a>	Liefert den Namen.
<a href="#">GetText [► 34]</a>	Liefert den Text.
<a href="#">Release [► 32]</a>	Löscht den internen Speicher des Bausteins

#### Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4026.0	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>=3.3.7.0)

### 3.1.9.1 Release



Diese Methode löscht den internen Speicher des Bausteins.

#### Syntax

METHOD Release : HRESULT

 **Ausgänge**

Name	Typ	Beschreibung
Release	HRESULT	Liefert den Erfolg, ob der Speicher aufgeräumt wurde.

**3.1.9.2 GetComment**



Diese Methode liefert den Kommentar.

**Syntax**

```
METHOD GetComment : BOOL
VAR_INPUT
    sResult      : REFERENCE TO STRING;
    nResult      : UDINT; // buffer size in bytes
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Referenz auf einen String, in dem das Ergebnis abgelegt werden soll.
nResult	UDINT	Länge des sResult

 **Ausgänge**

Name	Typ	Beschreibung
GetCommend	BOOL	Liefert den Erfolg, ob ein Kommentar bereitgestellt wurde.

**3.1.9.3 GetName**



Diese Methode liefert den Namen.

**Syntax**

```
METHOD GetName : BOOL
VAR_INPUT
    sResult      : REFERENCE TO STRING;
    nResult      : UDINT; // buffer size in bytes
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Referenz auf einen String, in dem das Ergebnis abgelegt werden soll.
nResult	UDINT	Länge des sResult

 **Ausgänge**

Name	Typ	Beschreibung
GetName	BOOL	Liefert den Erfolg, ob ein Name bereitgestellt wurde.

**3.1.9.4      GetText**



Diese Methode liefert den Text.

**Syntax**

```
METHOD GetText : BOOL
VAR_INPUT
    sResult      : REFERENCE TO STRING;
    nResult      : UDINT; // buffer size in bytes
END_VAR
```

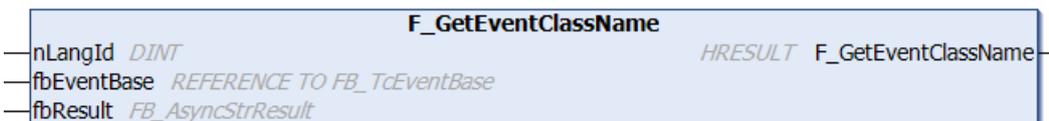
 **Eingänge**

Name	Typ	Beschreibung
sResult	REFERENCE TO STRING	Referenz auf einen String, in dem das Ergebnis abgelegt werden soll.
nResult	UDINT	Länge des sResult

 **Ausgänge**

Name	Typ	Beschreibung
GetText	BOOL	Liefert den Erfolg, ob ein Text bereitgestellt wurde.

**3.1.10      F\_GetEventClassName**



Die Funktion triggert die asynchrone Anfrage des Namens einer Ereignisklasse.

**Syntax**

Definition:

```
FUNCTION F_GetEventClassName : HRESULT
VAR_INPUT
    nLangId      : DINT;
    fbEventBase  : REFERENCE TO FB_TcEventBase;
END_VAR
VAR_IN_OUT
    fbResult     : FB_AsyncStrResult;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nLangId	DINT	Spezifiziert die Sprach-ID Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ...
fbEventBase	REFERENCE TO FB_TcEventBase [ <a href="#">▶ 72</a> ]	Angabe eines Event/Alarm/Message-Objektes.

 /  **Ein-/Ausgänge**

Name	Typ	Beschreibung
fbResult	FB_AsyncStrResult [ <a href="#">▶ 11</a> ]	Angabe einer Bausteininstanz, um die asynchrone Textanfrage zu verfolgen.

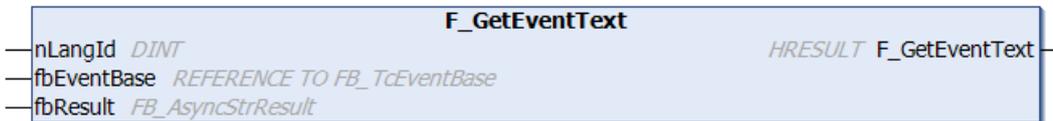
 **Rückgabewert**

Name	Typ	Beschreibung
F_GetEventClassName	HRESULT	Liefert mögliche Fehlerinformationen.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 3.1.11 F\_GetEventText



Die Funktion triggert die asynchrone Anfrage eines Ereignistextes.

**Syntax**

Definition:

```

FUNCTION F_GetEventText : HRESULT
VAR_INPUT
    nLangId      : DINT;
    fbEventBase : REFERENCE TO FB_TcEventBase;
END_VAR
VAR_IN_OUT
    fbResult : FB_AsyncStrResult;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
nLangId	DINT	Spezifiziert die Sprach-ID Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ...
fbEventBase	REFERENCE TO FB_TcEventBase [ <a href="#">▶ 72</a> ]	Angabe eines Event/Alarm/Message-Objektes.

 /  **Ein-/Ausgänge**

Name	Typ	Beschreibung
fbResult	FB_AsyncStrResult [ <a href="#">▶ 11</a> ]	Angabe einer Bausteininstanz, um die asynchrone Textanfrage zu verfolgen.

 **Rückgabewert**

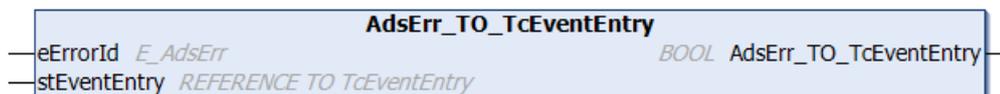
Name	Typ	Beschreibung
F_GetEventText	HRESULT	Liefert mögliche Fehlerinformationen.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

## 3.2 EventEntry-Konvertierung

### 3.2.1 AdsErr\_TO\_TcEventEntry



Diese Funktion konvertiert einen Standard-ADS-Fehler in ein TcEventEntry.

**Syntax**

Definition:

```
FUNCTION AdsErr_TO_TcEventEntry : BOOL
VAR_INPUT
    eErrorId      : E_AdsErr;
    stEventEntry  : REFERENCE TO TcEventEntry;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
eErrorId	E_AdsErr	Zu konvertierender Fehlercode.
stEventEntry	REFERENCE TO TcEventEntry [ <a href="#">▶ 113</a> ]	Gibt die resultierende Ereignisdefinition aus.

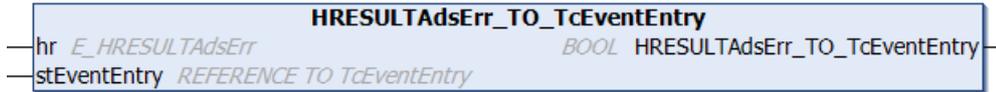
 **Rückgabewert**

Name	Typ	Beschreibung
AdsErr_TO_TcEventEntry	BOOL	Liefert TRUE, wenn die Konvertierung erfolgreich durchgeführt wurde.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 3.2.2 HRESULTAdsErr\_TO\_TcEventEntry



Diese Funktion konvertiert einen Standard-ADS-Fehler (HRESULT) in ein TcEventEntry.

#### Syntax

Definition:

```
FUNCTION HRESULTAdsErr_TO_TcEventEntry : BOOL
VAR_INPUT
    hr          : E_HRESULTAdsErr;
    stEventEntry : REFERENCE TO TcEventEntry;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
hr	E_HRESULTAdsErr	Zu konvertierender Fehlercode.
stEventEntry	REFERENCE TO TcEventEntry <a href="#">▶ 113</a>	Gibt die resultierende Ereignisdefinition aus.

#### Rückgabewert

Name	Typ	Beschreibung
HRESULTAdsErr_TO_TcEventEntry	BOOL	Liefert TRUE, wenn die Konvertierung erfolgreich durchgeführt wurde. Der Aufruf schlägt fehl, wenn der Facility Code im angegebenen HRESULT unbekannt ist.

#### Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 3.2.3 TcEventEntry\_TO\_AdsErr



Diese Funktion konvertiert ein TcEventEntry in einen Standard-ADS-Fehler.

#### Syntax

Definition:

```
FUNCTION TcEventEntry_TO_AdsErr : BOOL
VAR_INPUT
    stEventEntry : TcEventEntry;
    eErrorId     : REFERENCE TO E_AdsErr;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
stEventEntry	TcEventEntry [ <a href="#">▶ 113</a> ]	Zu konvertierende Ereignisdefinition.
eErrorId	REFERENCE TO E_AdsErr	Gibt den resultierenden Fehlercode aus.

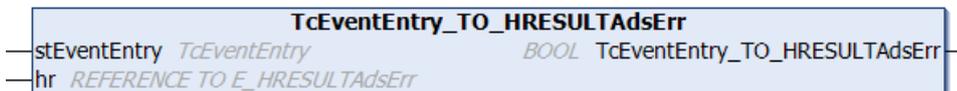
 **Rückgabewert**

Name	Typ	Beschreibung
TcEventEntry_TO_AdsErr	BOOL	Liefert TRUE, wenn die Konvertierung erfolgreich durchgeführt wurde und FALSE, wenn die Ereignisklasse unbekannt ist.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 3.2.4 TcEventEntry\_TO\_HRESULTAdsErr



Diese Funktion konvertiert ein TcEventEntry in einen Standard-ADS-Fehler (HRESULT).

**Syntax**

Definition:

```
FUNCTION TcEventEntry_TO_HRESULTAdsErr : BOOL
VAR_INPUT
    stEventEntry : TcEventEntry;
    hr           : REFERENCE TO E_HRESULTAdsErr;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
stEventEntry	TcEventEntry [ <a href="#">▶ 113</a> ]	Zu konvertierende Ereignisdefinition.
hr	REFERENCE TO E_HRESULTAdsErr	Gibt den resultierenden Fehlercode aus.

 **Rückgabewert**

Name	Typ	Beschreibung
TcEventEntry_TO_HRESULTAdsErr	BOOL	Liefert TRUE, wenn die Konvertierung erfolgreich durchgeführt wurde und FALSE, wenn die Ereignisklasse unbekannt ist.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 3.3 Filter

Die Filterfunktionalität wird an unterschiedlichen Stellen verwendet. Ein Beispiel, welches die Verwendungsmöglichkeiten beschreibt, ist das [Beispiel Filter \[▶ 119\]](#).

#### 3.3.1 FB\_TcClearLoggedEventsSettings



Dieser Funktionsbaustein bietet die Funktionalität, um festzulegen, welche Ereignisse aus dem Cache entfernt werden sollen.

#### Syntax

Definition:

```
FUNCTION_BLOCK FB_TcClearLoggedEventsSettings IMPLEMENTS I_TcClearLoggedEventsSettings
```

#### Methoden

Name	Definitionsort	Beschreibung
<a href="#">AddFilter [▶ 39]</a>	Lokal	Methode, um einen Filter hinzuzufügen. Liefert bei Erfolg S_OK.
<a href="#">Clear [▶ 40]</a>	Lokal	Methode zum Löschen der Einstellungen. Liefert bei Erfolg S_OK.
<a href="#">SetLimit [▶ 40]</a>	Lokal	Gibt die Anzahl der zu löschenden Ereignisse an. Die Begrenzung wird nach dem Sortieren und Filtern angewendet. Liefert bei Erfolg S_OK.
<a href="#">SetSorting [▶ 40]</a>	Lokal	Legt die Sortierung für die Abfrage fest. Liefert bei Erfolg S_OK.

#### Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.17	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>= v3.1.27.0)

#### 3.3.1.1 AddFilter



Methode, um einen Filter hinzuzufügen.

#### Eingänge

Name	Typ	Beschreibung
ipEventFilter	I_TcEventFilter	Instanz des zu nutzenden Filters

#### Rückgabewerte

Name	Typ	Beschreibung
AddFilter	HRESULT	Liefert bei Erfolg S_OK.

### 3.3.1.2 Clear

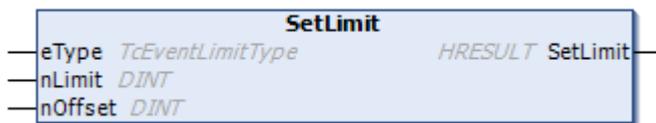


Methode zum Löschen der Einstellungen.

Rückgabewerte

Name	Typ	Beschreibung
Clear	HRESULT	Liefert bei Erfolg S_OK.

### 3.3.1.3 SetLimit



Gibt die Anzahl der zu löschenden Ereignisse an. Die Begrenzung wird nach dem Sortieren und Filtern angewendet.

Eingänge

Name	Typ	Beschreibung
eType	TcEventLimitType	Bestimmt die Referenz, ob die Begrenzung für die ersten oder letzten Ereignisse gilt.
nLimit	DINT	Spezifiziert die Anzahl (-1 = no limit)
nOffset	DINT	Optional. Definiert, wie viele Einträge übersprungen werden sollen.

Rückgabewerte

Name	Typ	Beschreibung
SetLimit	HRESULT	Liefert bei Erfolg S_OK.

### 3.3.1.4 SetSorting



Legt die Sortierung für die Abfrage fest.

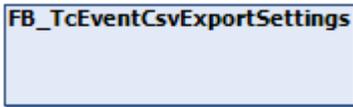
Eingänge

Name	Typ	Beschreibung
eField	TcEventField	Eigenschaft, die für die Sortierung verwendet werden soll.
eOrder	TcEventSortOrder	Definiert die Sortierreihenfolge.

 Rückgabewerte

Name	Typ	Beschreibung
SetSorting	HRESULT	Liefert bei Erfolg S_OK.

### 3.3.2 FB\_TcEventCsvExportSettings



Bietet die Funktionalität, den csv-Export zu spezifizieren.

#### Syntax

Definition:

```
FUNCTION_BLOCK FB_TcEventCsvExportSettings EXTENDS FB_TcEventExportSettings IMPLEMENTS I_TcEventCsvExportSettings
```

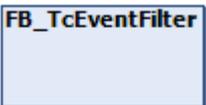
 Methoden

Name	Typ	Beschreibung
bWithHeader	BOOL	Bestimmt, ob eine Kopfzeile erstellt werden soll. Standard: True
nLangId	DINT	Bestimmt die Standard-Kennung der Exportsprache. Standard: 1033
sDelimiter	STRING	Definiert das CSV-Begrenzungszeichen. Standard: Semicolon [;]

#### Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.17	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>= v3.1.27.0)

### 3.3.3 FB\_TcEventFilter



Stellt die Funktionalität zur Verfügung, um einen Ereignisfilter zu spezifizieren.

Die Filter werden über eine fließende Schnittstelle in Anlehnung an eine strukturierte Abfragesprache gegeben. Diese beschreibt, welche Nachrichten zutreffen sollen.

- Bedingungen können durch `.AND_OP()` und `.OR_OP()` verknüpft werden.
- Bedingungen können durch `.NOT_OP()` negiert werden.
- Bedingungen können durch Eigenschaften wie `.isAlarm()` oder beispielsweise `.EventClass.EqualsTo(<EventClass>)` definiert werden. Eine vollständige Liste der Eigenschaften befindet sich in der API-Dokumentation.
- Eine Gruppierung kann durch `.FilterExpression(<SubCondition>)` formuliert werden. Die `<SubCondition>` ist selbst wieder ein `FB_TcEventFilter` bzw. `ITcEventFilter`.

Nachdem ein Filter zusammengestellt wurde, wird er angewendet. Für das Empfangen von Nachrichten wird er beispielsweise über `FB_ListenerBase2.subscribe()` einem Empfänger zugeordnet. Der `FB_ListenerBase2` übernimmt hierdurch den Filter und gibt einen entsprechenden Rückgabewert, welcher hier beschrieben ist. Eine Änderung des Filters durch erneuten `FB_ListenerBase2.subscribe()` vornehmen.

Es können maximal 255 Filterbedingungen gesetzt werden – danach wird als Fehlermeldung ein `ADS_NOMOREHDL` rückgegeben.

### Beispiel

Beispielsweise kann ein Filter auf diese Art zusammengestellt werden:

```
fbFilter.Severity.GreaterThan (TcEventSeverity.Error).AND_OP().Source.Name.Like('%Main%');
```

Das [Beispiel Filter \[► 119\]](#) zeigt die Verwendung.

### EtherCAT Filter

Der Empfang der EtherCAT Emergency Nachrichten ist ähnlich zu dem zuvor beschriebenen Mechanismus aufgebaut. Der Einstiegspunkt in den verketteten Methodenaufrufen ist `.EtherCATDevice()`, welches als erstes die direkte Anfrage bietet, ob es von einem EtherCAT Gerät abgesendet wurde (`IsEtherCATDevice()`). Von hier aus kann auf den Hersteller (`.VendorId()`), den ProductCode (`.ProductCode()`) sowie die Revision (`.RevisionNo()`) gefiltert werden.

### Syntax

Definition:

```
FUNCTION_BLOCK FB_TcEventFilter IMPLEMENTS I_TcEventFilter, I_TcExpressionBase
```

### Methoden

Name	Definitionsart	Beschreibung
<a href="#">Clear [► 43]</a>	<code>I_TcEventFilter</code>	Löscht den bisherige Filterausdruck.
<a href="#">FilterExpression [► 43]</a>	<code>I_TcExpressionBase</code>	Angabe einer unterlagerten Filterdefinition.
<a href="#">IsAlarm [► 44]</a>	<code>I_TcExpressionBase</code>	Überprüfung, ob es sich um einen Alarm handelt.
<a href="#">IsMessage [► 44]</a>	<code>I_TcExpressionBase</code>	Überprüfung, ob es sich um eine Nachricht handelt.
<a href="#">NOT_OP [► 44]</a>	<code>I_TcExpressionBase</code>	Negierung der folgenden Aussage.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
AlarmState	I_TcAlarmFilterExpression	Get	Abgleich mit einem AlarmState
EtherCATDevice	I_TcEtherCATDeviceExpression	Get	Abgleich, ob es sich um ein EtherCAT Gerät als Quelle handelt.
EventClass	I_TcGuidCompare	Get	Abgleich mit einer EventClass
EventId	I_TcUDIntCompare	Get	Abgleich mit einer EventId
JsonAttribute	I_TcJsonAttributeExpression	Get	Abgleich mit dem JsonAttribut
Severity	I_TcSeverityCompare	Get	Abgleich mit der Serverity
Source	I_TcSourceInfoExpression	Get	Abgleich mit der Quelle
TimeCleared	I_TcULIntCompare	Get	Abgleich des Clear-Zeitpunkts (nur bei Alarm)
TimeConfirmed	I_TcULIntCompare	Get	Abgleich des Confirm-Zeitpunkts (nur bei Alarm mit Quittierung)
TimeRaised	I_TcULIntCompare	Get	Abgleich des Absenders (bei Nachrichten) bzw. des Raised-Zeitpunktes (bei Alarmen)

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.17	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>= v3.1.27.0)

**3.3.3.1 Clear**



Die Methode löscht den bisherige Filterausdruck.

 **Rückgabewerte**

Name	Typ	Beschreibung
Clear	I_TcBinaryExpression	Verknüpfungspunkt

**3.3.3.2 FilterExpression**



Die Methode gibt eine unterlagerte Filterdefinition an.

 **Eingänge**

Name	Typ	Beschreibung
ipExpression	I_TcEventFilterBase	Verknüpfungspunkt

### 📌 Rückgabewerte

Name	Typ	Beschreibung
FilterExpression	I_TcLogicalExpression	Verknüpfungspunkt

#### 3.3.3.3 IsAlarm

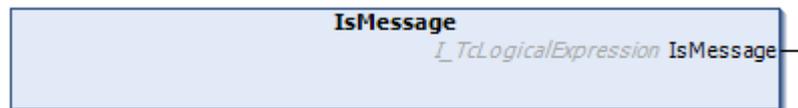


Die Methode überprüft, ob es sich um einen Alarm handelt.

### 📌 Rückgabewerte

Name	Typ	Beschreibung
IsAlarm	I_TcLogicalExpression	Verknüpfungspunkt

#### 3.3.3.4 IsMessage



Die Methode überprüft, ob es sich um eine Nachricht handelt.

### 📌 Rückgabewerte

Name	Typ	Beschreibung
IsMessage	I_TcLogicalExpression	Verknüpfungspunkt

#### 3.3.3.5 NOT\_OP



Die Methode negiert die folgende Aussage.

### 📌 Rückgabewerte

Name	Typ	Beschreibung
NOT_OP	I_TcBinaryExpression	Verknüpfungspunkt

## 3.4 RemoteEventLogger

Der Eventlogger bietet die Möglichkeit auch auf entfernte Systeme zuzugreifen, wofür hier die nötigen Funktionsbausteine beschrieben sind.

Zusätzlich besitzen Funktionsbausteine aus dem Bereich [Asynchrone Textanfragen](#) [11] eine Option um auch Texte auf einem entfernten System abzufragen.

Die Verwendung ist im Bereich Remote Zugriff auf den Eventlogger mittels PLC beschrieben.

### 3.4.1 FB\_RemoteListenerBase



Der Funktionsbaustein dient als Basisimplementierung eines Ereignisbeobachters eines entfernten Systems. Durch das Überschreiben der ereignisgesteuerten Methoden können neue Nachrichten und Zustandsänderungen von Alarmen erkannt werden.

Dieser Funktionsbaustein bietet den Zugriff auf den Eventlogger eines entfernten Systems und kann verwendet werden, um dort Ereignisse abzusenden oder von dort zu empfangen.

#### Syntax

```
FUNCTION_BLOCK FB_RemoteListenerBase IMPLEMENTS I_RemoteListener
```

#### Methoden

Name	Beschreibung
<a href="#">Execute [▶ 46]</a>	Muss zyklisch aufgerufen werden, damit die Ereignis-Queue abgearbeitet werden kann.
<a href="#">Subscribe [▶ 49]</a>	Meldet Benachrichtigungen an.
<a href="#">Unsubscribe [▶ 50]</a>	Meldet Benachrichtigungen ab.

#### ⚡ Ereignisgesteuerte Methoden

Name	Beschreibung
<a href="#">OnAlarmCleared [▶ 46]</a>	Wird aufgerufen, wenn der Zustand eines Alarms von „Raised“ nach „Clear“ wechselt.
<a href="#">OnAlarmConfirmed [▶ 46]</a>	Wird aufgerufen, wenn ein Alarm bestätigt wurde.
<a href="#">OnAlarmDisposed [▶ 47]</a>	Wird aufgerufen, wenn eine Alarminstanz wieder frei gegeben wurde.
<a href="#">OnAlarmRaised [▶ 47]</a>	Wird aufgerufen, wenn der Zustand eines Alarms von „Clear“ nach „Raised“ wechselt.
<a href="#">OnConnectionStateChanged [▶ 47]</a>	Wird aufgerufen, wenn die Verbindung zum entfernten System ihren Zustand ändert.
<a href="#">OnDatabaseChanged [▶ 48]</a>	Wird aufgerufen, wenn die Datenbank sich geändert hat, Z.B. wenn ein ClearLoggedEvents() aufgerufen wurde
<a href="#">OnEventLoggerError [▶ 48]</a>	Wird aufgerufen, wenn bei der Kommunikation ein ADS-Fehler auftritt.
<a href="#">OnMessageSent [▶ 49]</a>	Wird aufgerufen, wenn eine Nachricht abgeschickt wurde.

#### 📄 Eigenschaften

Name	Typ	Zugriff	Beschreibung
bSubscribed	BOOL	Get	Liefert TRUE, wenn der Funktionsbaustein Benachrichtigungen angemeldet hat und die Ereignisbeobachtung aktiv ist.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4026.0	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>=3.3.7.0)

**3.4.1.1 Execute**



Die Methode muss zyklisch aufgerufen werden, damit die Ereignis-Queue abgearbeitet werden kann.

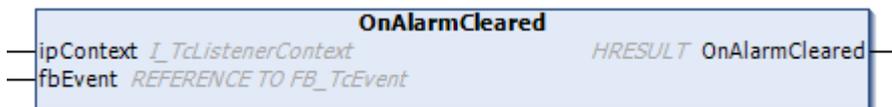
**Syntax**

```
METHOD Execute : HRESULT
```

**Rückgabewert**

Name	Typ	Beschreibung
Execute	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

**3.4.1.2 OnAlarmCleared**



Diese Methode wird aufgerufen, wenn der Zustand eines Alarms von Raised nach Clear wechselt.

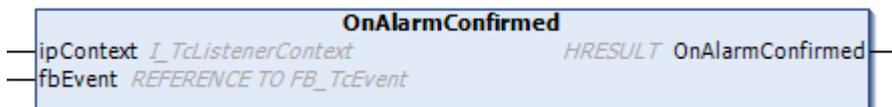
**Syntax**

```
METHOD OnAlarmCleared : HRESULT
VAR_INPUT
    ipContext      : I_TcListenerContext;
    fbEvent       : REFERENCE TO FB_TcEvent;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
ipContext	I_TcListenerContext	Context der unterschiedlichen Eventlogger
fbEvent	REFERENCE TO FB_TcEvent	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z. B. per Zuweisung kopiert werden.

**3.4.1.3 OnAlarmConfirmed**



Diese Methode wird aufgerufen, wenn ein Alarm bestätigt wurde.

**Syntax**

```
METHOD OnAlarmConfirmed : HRESULT
VAR_INPUT
    ipContext      : I_TcListenerContext;
    fbEvent       : REFERENCE TO FB_TcEvent;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
ipContext	I_TcListenerContext	Context der unterschiedlichen Eventlogger
fbEvent	REFERENCE TO FB_TcEvent	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z. B. per Zuweisung kopiert werden.

**3.4.1.4 OnAlarmDisposed**



Diese Methode wird aufgerufen, wenn eine Alarminstanz wieder frei gegeben wurde.

**Syntax**

```

METHOD OnAlarmDisposed : HRESULT
VAR_INPUT
    ipContext          : I_TcListenerContext;
    fbEvent            : REFERENCE TO FB_TcEvent;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
ipContext	I_TcListenerContext	Context der unterschiedlichen Eventlogger
fbEvent	REFERENCE TO FB_TcEvent	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z. B. per Zuweisung kopiert werden.

**3.4.1.5 OnAlarmRaised**



Diese Methode wird aufgerufen, wenn der Zustand eines Alarms von Clear nach Raised wechselt.

**Syntax**

```

METHOD OnAlarmRaised : HRESULT
VAR_INPUT
    ipContext          : I_TcListenerContext;
    fbEvent            : REFERENCE TO FB_TcEvent;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
ipContext	I_TcListenerContext	Context der unterschiedlichen Eventlogger
fbEvent	REFERENCE TO FB_TcEvent	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z. B. per Zuweisung kopiert werden.

**3.4.1.6 OnConnectionStateChanged**



Diese Methode wird aufgerufen, wenn die Verbindung zum entfernten System ihren Zustand ändert.

**Syntax**

```
METHOD OnConnectionStateChanged : HRESULT
VAR_INPUT
    ipContext          : I_TcListenerContext;
    eReason            : TcRemoteConnectionChangeReason;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
ipContext	I_TcListenerContext	Context der unterschiedlichen Eventlogger
eReason	TcRemoteConnectionChangeReason	Die Art der Änderung des Verbindungsstatus.

**3.4.1.7 OnDatabaseChanged**

```

OnDatabaseChanged
— ipContext I_TcListenerContext          HRESULT OnDatabaseChanged
— eReason  TcDatabaseChangeReason
```

Diese Methode wird aufgerufen, wenn sich die Datenbank geändert hat, z. B. wenn ein ClearLoggedEvents() aufgerufen wurde.

**Syntax**

```
METHOD OnDatabaseChanged : HRESULT
VAR_INPUT
    ipContext          : I_TcListenerContext;
    eReason            : TcDatabaseChangeReason;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
ipContext	I_TcListenerContext	Context der unterschiedlichen Eventlogger
eReason	TcDatabaseChangeReason	Einer der Werte des ENUM: Undefined, ClearedByCommand, MaximumSizeReached, Deleted

**3.4.1.8 OnEventLoggerError**

```

OnEventLoggerError
— ipContext I_TcListenerContext          HRESULT OnEventLoggerError
— hr       HRESULT
```

Diese Methode wird aufgerufen, wenn bei der Kommunikation ein ADS-Fehler auftritt.

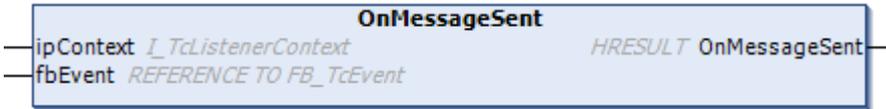
**Syntax**

```
METHOD OnEventLoggerError : HRESULT
VAR_INPUT
    ipContext          : I_TcListenerContext;
    hr                 : HRESULT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
ipContext	I_TcListenerContext	Context der unterschiedlichen Eventlogger
hr	HRESULT	Der aufgetretene Fehlercode.

**3.4.1.9 OnMessageSent**



Diese Methode wird aufgerufen, wenn eine Nachricht gesendet wurde.

**Syntax**

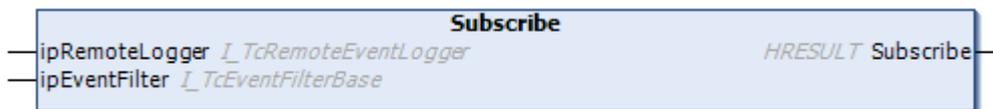
```

METHOD OnMessageSent : HRESULT
VAR_INPUT
    ipContext      : I_TcListenerContext;
    fbEvent        : REFERENCE TO FB_TcEvent;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
ipContext	I_TcListenerContext	Context der unterschiedlichen Eventlogger
fbEvent	REFERENCE TO FB_TcEvent	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z. B. per Zuweisung kopiert werden.

**3.4.1.10 Subscribe**



Die Methode meldet Benachrichtigungen an.

**Syntax**

```

METHOD Subscribe : HRESULT
VAR_INPUT
    ipRemoteLogger : I_TcRemoteEventLogger;
    ipEventFilter  : I_TcEventFilterBase; // optional
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
ipRemoteLogger	I_TcRemoteEventLogger	Referenz auf den Eventlogger, bei dem eine Benachrichtigung angemeldet werden soll.
ipEventFilter	I_TcEventFilterBase	Zeiger auf eine Instanz von <a href="#">FB_TcEventFilter</a> ( <a href="#">▶ 41</a> ), falls ein Filter aktiviert werden soll.

 **Rückgabewert**

Name	Typ	Beschreibung
Subscribe	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ADS_E_EXISTS, wenn der Beobachter bereits registriert ist. Liefert ansonsten ein HRESULT als Fehlercode

**3.4.1.11 Unsubscribe**



Mit dieser Methode wird der Beobachter abgemeldet.

**Syntax**

```
METHOD Unsubscribe : HRESULT
```

 **Rückgabewert**

Name	Typ	Beschreibung
Unsubscribe	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ADS_E_NOTFOUND, wenn der Beobachter nicht registriert war. Liefert ansonsten ein HRESULT als Fehlercode.

**3.4.2 FB\_TcRemoteEventLogger**



Dieser Funktionsbaustein stellt den TwinCAT 3 EventLogger für ein entferntes System dar.

**Syntax**

```
FUNCTION_BLOCK FB_RemoteEventLogger IMPLEMENTS I_RemoteEventLogger
```

 **Methoden**

Name	Beschreibung
ClearAlarm [▶ 51]s	Löscht aktive Alarmer.
ClearLoggedEvents [▶ 52]	Löscht protokollierte Ereignisse.
ConfirmAlarms [▶ 53]	Bestätigt Alarmer.
Connect [▶ 53]	Verbindet sich mit dem entfernten System.
Disconnect [▶ 54]	Löst die Verbindung mit dem entfernten System auf.
SendMessage [▶ 54]	Sendet eine Nachricht.
SendMessage2 [▶ 55]	Sendet eine Nachricht.
SendMessageEx [▶ 56]	Sendet eine Nachricht.
SendMessageEx2 [▶ 56]	Sendet eine Nachricht.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
hrConnect	HRESULT	Get	Verbindungsstatus zum entfernten System
hrInit	HRESULT	Get	Fehler, die während des Aufstarten auftreten.
sNetId	T_AmsNetId	Get	AmsNetId des entfernten Systems

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4026.0	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>=3.3.7.0)

**3.4.2.1 ClearAlarms**



Methode zum Löschen aktiver Alarmer. Gibt S\_OK zurück, wenn erfolgreich.

**Syntax**

```

METHOD ClearAlarms      : HRESULT
VAR_INPUT
    nTimeStamp          : ULINT := 0;
    bResetConfirmation  : BOOL := FALSE;
    ipFilter             : I_TcEventFilter;
END_VAR
    
```

**Eingänge**

Name	Typ	Beschreibung
nTimeStamp	ULINT	0 setzen, um die aktuelle Zeit automatisch zu erhalten. Initial: 0
bResetConfirmation	BOOL	Wenn TRUE und der Bestätigungsstatus WaitForConfirmation ist, wird der Bestätigungsstatus auf Reset gesetzt. Andernfalls wird der Bestätigungsstatus nicht geändert. Initial: FALSE
ipFilter	I_TcEventFilter	Angaben, welche Alarime gelöscht werden sollen, ansonsten werden alle ausgelösten Alarime gelöscht.

**Rückgabewert**

Name	Typ	Beschreibung
ClearAlarms	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

**3.4.2.2 ClearLoggedEvents**



Async-Methode zum Löschen von protokollierten Ereignissen. Gibt TRUE zurück, wenn die asynchrone Anfrage nicht mehr belegt ist.

**Syntax**

```
METHOD ClearLoggedEvents : HRESULT
VAR_INPUT
    ipClearSettings      : I_TcClearLoggedEventsSettings;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
ipClearSettings	I_TcClearLoggedEventsSettings	Optional (Andernfalls wird der ganze Cache geleert.)

**Rückgabewert**

Name	Typ	Beschreibung
ClearLoggedEvents	HRESULT	Liefert S_PENDING solange die Anfrage noch nicht abgeschlossen ist. Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

**Verwendung der Methode**

Diese Methode ist asynchron, d. h. sie benötigt mehrere Zyklen, um den Rückgabewert zu liefern. Folgende Verwendung ist vorgesehen:

```
IF bClearLoggedEvents THEN
    bClearLoggedEvents:= NOT (fbLogger.ClearLoggedEvents(0) = S_OK);
END_IF
```

### 3.4.2.3 ConfirmAlarms



Die Methode bestätigt Alarmer.

#### Syntax

```
METHOD ConfirmAlarms : HRESULT
VAR_INPUT
    nTimeStamp      : ULINT := 0;
    ipFilter        : I_TcEventFilter;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
nTimeStamp	ULINT	0 setzen, um die aktuelle Zeit automatisch zu erhalten. Initial: 0
ipFilter	I_TcEventFilter	Festlegen, welche Alarmer bestätigt werden sollen, ansonsten werden alle Alarmer mit dem Bestätigungsstatus WaitForConfirmation bestätigt.

#### Rückgabewert

Name	Typ	Beschreibung
ConfirmAlarms	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.4.2.4 Connect



Die Methode verbindet sich mit dem entfernten System.

#### Syntax

```
METHOD Connect : HRESULT
VAR_INPUT
    sNetId          : T_AmsNetId; (* AmsNetId of the remote TwinCAT system.*)
END_VAR
```

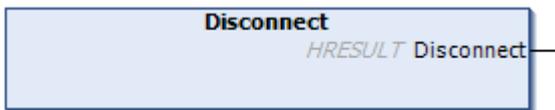
#### Eingänge

Name	Typ	Beschreibung
sNetId	T_AmsNetId	AmsNetId des entfernten Systems

#### Rückgabewert

Name	Typ	Beschreibung
Connect	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.4.2.5 Disconnect



Die Methode löst die Verbindung mit dem entfernten System auf.

#### Syntax

```
METHOD Disconnect : HRESULT
```

#### 📌 Rückgabewert

Name	Typ	Beschreibung
Disconnect	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.4.2.6 SendMessage



Diese Methode sendet eine Nachricht.

#### Syntax

```
METHOD SendMessage : HRESULT
VAR_INPUT
    eventClass      : GUID;
    nEventId       : UDINT;
    eSeverity      : TcEventSeverity;
    ipSourceInfo   : I_TcSourceInfo := 0; // optional
    nTimeStamp     : ULINT := 0; // set 0 to get the current time automatically
    ipArguments    : I_TcArguments := 0; // optional
END_VAR
```

#### 📌 Eingänge

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse
nEventId	UDINT	ID des Ereignisses
eSeverity	TcEventSeverity	Severity des Ereignisses
ipSourceInfo	I_TcSourceInfo	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcSourceInfo [▶ 93]</a> angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet. > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC)
ipArguments	I_TcArguments	Schnittstellenzeiger auf Argumente des Ereignisses. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcArguments [▶ 68]</a> angegeben werden.

 Rückgabewert

Name	Typ	Beschreibung
SendMessage	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.4.2.7 SendMessage2



Diese Methode sendet eine Nachricht.

#### Syntax

```

METHOD SendMessage2 : HRESULT
VAR_INPUT
    eventClass      : GUID;
    nEventId        : UDINT;
    eSeverity       : TcEventSeverity;
    ipSourceInfo    : I_TcSourceInfo := 0; // optional
    nTimeStamp      : ULINT := 0; // set 0 to get the current time automatically
    ipArguments     : I_TcArguments := 0; // optional
END_VAR
    
```

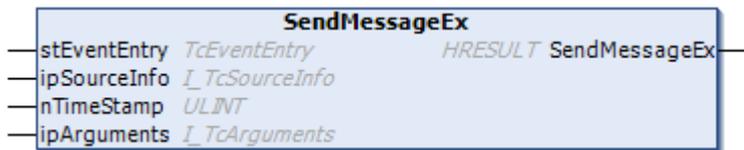
 Eingänge

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse
nEventId	UDINT	ID des Ereignisses.
eSeverity	TcEventSeverity	Severity des Ereignisses
ipSourceInfo	I_TcSourceInfo	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcSourceInfo</a> [93] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet. > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC)
ipArguments	I_TcArguments	Schnittstellenzeiger auf Argumente des Ereignisses. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcArguments</a> [68] angegeben werden.
sJsonAttribute	STRING	String als Json Repräsentation, der zusätzlich zum Ereignistext übertragen wird.

 Rückgabewert

Name	Typ	Beschreibung
SendMessage2	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.4.2.8 SendMessageEx



Diese Methode sendet eine Nachricht.

#### Syntax

```

METHOD SendMessageEx : HRESULT
VAR_INPUT
    stEventEntry      : TcEventEntry;
    ipSourceInfo      : I_TcSourceInfo := 0; // optional
    nTimeStamp        : ULINT := 0; // set 0 to get the current time automatically
    ipArguments       : I_TcArguments := 0; // optional
END_VAR
    
```

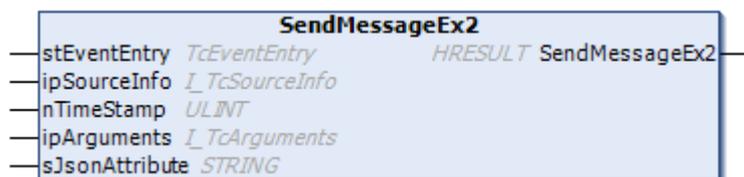
#### Eingänge

Name	Typ	Beschreibung
stEventEntry	TcEventEntry	Ereignisdefinition
ipSourceInfo	I_TcSourceInfo	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB TcSourceInfo [▶ 93]</a> angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet. > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC)
ipArguments	I_TcArguments	Schnittstellenzeiger auf Argumente des Ereignisses. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB TcArguments [▶ 68]</a> angegeben werden.

#### Rückgabewert

Name	Typ	Beschreibung
SendMessageEx	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.4.2.9 SendMessageEx2



Diese Methode sendet eine Nachricht.

#### Syntax

```

METHOD SendMessageEx : HRESULT
VAR_INPUT
    stEventEntry      : TcEventEntry;
    ipSourceInfo      : I_TcSourceInfo := 0; // optional
    nTimeStamp        : ULINT := 0; // set 0 to get the current time automatically
    ipArguments       : I_TcArguments := 0; // optional
    sJsonAttribute    : STRING;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
stEventEntry	TcEventEntry	Ereignisdefinition
ipSourceInfo	I_TcSourceInfo	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcSourceInfo</a> [ <a href="#">▶ 93</a> ] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet. > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC)
ipArguments	I_TcArguments	Schnittstellenzeiger auf Argumente des Ereignisses. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcArguments</a> [ <a href="#">▶ 68</a> ] angegeben werden.
sJsonAttribute	STRING	String als Json Repräsentation, der zusätzlich zum Ereignistext übertragen wird.

 **Rückgabewert**

Name	Typ	Beschreibung
SendMessageEx 2	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.5 FB\_ListenerBase2



Der Funktionsbaustein dient als Basisimplementierung eines Ereignisbeobachters.

Durch das Überschreiben der ereignisgesteuerten Methoden können neue Nachrichten und Zustandsänderungen von Alarmen erkannt werden.

**Syntax**

Definition:

```
FUNCTION_BLOCK FB_ListenerBase2 IMPLEMENTS I_Listener2
```

 **Methoden**

Name	Definitionsort	Beschreibung
<a href="#">Execute</a> [ <a href="#">▶ 58</a> ]	Lokal	Muss zyklisch aufgerufen werden, damit die Ereignis-Queue abgearbeitet werden kann.
<a href="#">Subscribe</a> [ <a href="#">▶ 60</a> ]	Lokal	Meldet Benachrichtigungen an.
<a href="#">Unsubscribe</a> [ <a href="#">▶ 61</a> ]	Lokal	Meldet Benachrichtigungen ab.

 Ereignisgesteuerte Methoden (Callback-Methoden)

Name	Definitionsort	Beschreibung
<a href="#">OnAlarmCleared</a> [▶ 58]	I_Listener2	Wird aufgerufen, wenn der Zustand eines Alarms von „Raised“ nach „Clear“ wechselt.
<a href="#">OnAlarmConfirmed</a> [▶ 59]	I_Listener2	Wird aufgerufen, wenn ein Alarm bestätigt wurde.
<a href="#">OnAlarmDisposed</a> [▶ 59]	I_Listener2	Wird aufgerufen, wenn eine Alarminstanz wieder frei gegeben wurde.
<a href="#">OnAlarmRaised</a> [▶ 60]	I_Listener2	Wird aufgerufen, wenn der Zustand eines Alarms von „Clear“ nach „Raised“ wechselt.
<a href="#">OnMessageSent</a> [▶ 60]	I_Listener2	Wird aufgerufen, wenn eine Nachricht abgeschickt wurde.

 Eigenschaften

Name	Typ	Zugriff	Definitionsort	Beschreibung
bSubscribed	BOOL	Get	Lokal	Liefert TRUE, wenn der Funktionsbaustein Benachrichtigungen angemeldet hat und die Ereignisbeobachtung aktiv ist.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.17	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>= v3.1.27.0)

### 3.5.1 Execute



Diese Methode muss zyklisch aufgerufen werden, damit die Ereignis-Queue abgearbeitet werden kann.

**Syntax**

METHOD Execute : HRESULT

 Rückgabewert

Name	Typ	Beschreibung
Execute	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode

### 3.5.2 OnAlarmCleared



Diese Methode wird aufgerufen, wenn der Zustand eines Alarms von Raised nach Clear wechselt.

**Syntax**

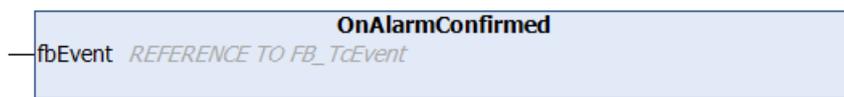
```
METHOD OnAlarmCleared : HRESULT
VAR_INPUT
    fbEvent : REFERENCE TO FB_TcEvent;
END_VAR
```

Liefert die Implementierung der Callback-Methode einen Returncode <> S\_OK zurück, so werden weitere Callback-Aufrufe bis zur nächsten Ausführung pausiert.

 **Eingänge**

Name	Typ	Beschreibung
fbEvent	REFERENCE TO <a href="#">FB_TcEvent</a> [▶ 70]	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z.B. per Zuweisung kopiert werden.

**3.5.3 OnAlarmConfirmed**



Diese Methode wird aufgerufen, wenn ein Alarm bestätigt wurde.

**Syntax**

```
METHOD OnAlarmConfirmed : HRESULT
VAR_INPUT
    fbEvent : REFERENCE TO FB_TcEvent;
END_VAR
```

Liefert die Implementierung der Callback-Methode einen Returncode <> S\_OK zurück, so werden weitere Callback-Aufrufe bis zur nächsten Ausführung pausiert.

 **Eingänge**

Name	Typ	Beschreibung
fbEvent	REFERENCE TO <a href="#">FB_TcEvent</a> [▶ 70]	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z.B. per Zuweisung kopiert werden.

**3.5.4 OnAlarmDisposed**



Diese Methode wird aufgerufen, wenn eine Alarminstanz wieder frei gegeben wurde.

**Syntax**

```
METHOD OnAlarmConfirmed : HRESULT
VAR_INPUT
    fbEvent : REFERENCE TO FB_TcEvent;
END_VAR
```

Liefert die Implementierung der Callback-Methode einen Returncode <> S\_OK zurück, so werden weitere Callback-Aufrufe bis zur nächsten Ausführung pausiert.

 **Eingänge**

Name	Typ	Beschreibung
fbEvent	REFERENCE TO <a href="#">FB_TcEvent</a> [▶ 70]	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z.B. per Zuweisung kopiert werden.

### 3.5.5 OnAlarmRaised



Diese Methode wird aufgerufen, wenn der Zustand eines Alarms von Clear nach Raised wechselt.

#### Syntax

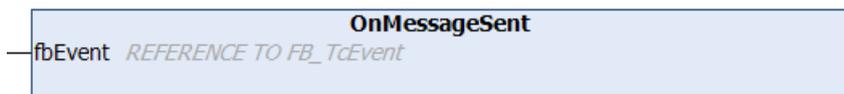
```
METHOD OnAlarmRaised : HRESULT
VAR_INPUT
    fbEvent : REFERENCE TO FB_TcEvent;
END_VAR
```

Liefert die Implementierung der Callback-Methode einen Returncode <> S\_OK zurück, so werden weitere Callback-Aufrufe bis zur nächsten Ausführung pausiert.

#### Eingänge

Name	Typ	Beschreibung
fbEvent	REFERENCE TO <a href="#">FB_TcEvent</a> [ <a href="#">▶ 70</a> ]	Referenz auf den aufgetretenen Alarm. Diese Referenz darf nicht z.B. per Zuweisung kopiert werden.

### 3.5.6 OnMessageSent



Diese Methode wird aufgerufen, wenn eine Nachricht gesendet wurde.

#### Syntax

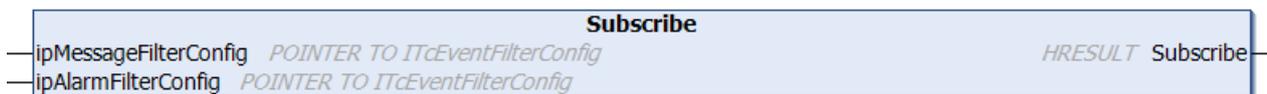
```
METHOD OnMessageSent : HRESULT
VAR_INPUT
    fbEvent : REFERENCE TO FB_TcEvent;
END_VAR
```

Liefert die Implementierung der Callback-Methode einen Returncode <> S\_OK zurück, so werden weitere Callback-Aufrufe bis zur nächsten Ausführung pausiert.

#### Eingänge

Name	Typ	Beschreibung
fbEvent	REFERENCE TO <a href="#">FB_TcEvent</a> [ <a href="#">▶ 70</a> ]	Referenz auf das aufgetretene Ereignis. Diese Referenz darf nicht z.B. per Zuweisung kopiert werden.

### 3.5.7 Subscribe



Mit dieser Methode wird der Beobachter für Benachrichtigungen angemeldet.

#### Syntax

```
METHOD Subscribe : HRESULT
VAR_INPUT
    ipMessageFilterConfig : POINTER TO ITcEventFilterConfig;
    ipAlarmFilterConfig : POINTER TO ITcEventFilterConfig;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
ipMessageFilterConfig	POINTER TO ITcEventFilterConfig	Zeiger auf ITcEventFilterConfig, wenn ein Filter aktiviert werden soll.
ipAlarmFilterConfig	POINTER TO ITcEventFilterConfig	Zeiger auf ITcEventFilterConfig, wenn ein Filter aktiviert werden soll.

**Rückgabewert**

Name	Typ	Beschreibung
Subscribe	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ADS_E_EXISTS, wenn der Beobachter bereits registriert ist. Liefert ansonsten ein HRESULT als Fehlercode.

### 3.5.8 Subscribe2



Diese Methode meldet Benachrichtigungen an.

**Syntax**

```
METHOD Subscribe2 : HRESULT
```

**Eingang**

Name	Typ	Beschreibung
ipEventFilter	I_TcEventFilterBase	Zeiger auf eine Instanz von <a href="#">FB_TcEventFilter</a> [► 41], falls ein Filter aktiviert werden soll.

**Rückgabewert**

Name	Typ	Beschreibung
Subscribe2	HRESULT	Liefert bei Erfolg S_OK.

### 3.5.9 Unsubscribe



Mit dieser Methode wird der Beobachter abgemeldet.

**Syntax**

```
METHOD Unsubscribe : HRESULT
```

### Rückgabewert

Name	Typ	Beschreibung
Unsubscribe	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ADS_E_NOTFOUND, wenn der Beobachter nicht registriert war. Liefert ansonsten ein HRESULT als Fehlercode.

## 3.6 FB\_TcAlarm



Dieser Funktionsbaustein repräsentiert einen Alarm des TwinCAT 3 EventLogger.

### Syntax

Definition:

```
FUNCTION_BLOCK FB_TcAlarm EXTENDS FB_TcEventBase
```

### Vererbungshierarchie

FB\_TcEventBase [[▶ 72](#)]

FB\_TcAlarm

☰ Methoden

Name	Definitionsort	Beschreibung
<a href="#">EqualsTo [▶ 73]</a>	Geerbt von <a href="#">FB_TcEventBase [▶ 72]</a>	Vergleicht das Ereignis mit einer anderen Instanz.
<a href="#">EqualsToEventClass [▶ 74]</a>	Geerbt von <a href="#">FB_TcEventBase [▶ 72]</a>	Vergleicht die Ereignisklasse des Ereignisses mit einer anderen Ereignisklasse.
<a href="#">EqualsToEventEntry [▶ 74]</a>	Geerbt von <a href="#">FB_TcEventBase [▶ 72]</a>	Vergleicht die Ereignisklasse, die Event-ID und die Severity des Ereignisses mit denen eines anderen Ereignisses.
<a href="#">EqualsToEventEntryEx [▶ 75]</a>	Geerbt von <a href="#">FB_TcEventBase [▶ 72]</a>	Vergleicht die Ereignisdefinition des Ereignisses mit einer anderen Ereignisdefinition.
<a href="#">GetJsonAttribute [▶ 75]</a>	Geerbt von <a href="#">FB_TcEventBase [▶ 72]</a>	Liefert das Json-Attribut.
<a href="#">Release [▶ 76]</a>	Geerbt von <a href="#">FB_TcEventBase [▶ 72]</a>	Gibt die vom EventLogger erstellte Instanz wieder frei.
<a href="#">RequestEventClassName [▶ 76]</a>	Geerbt von <a href="#">FB_TcEventBase [▶ 72]</a>	Fragt den Namen der Ereignisklasse an.
<a href="#">RequestEventText [▶ 77]</a>	Geerbt von <a href="#">FB_TcEventBase [▶ 72]</a>	Liefert den Text zum Ereignis.
<a href="#">Clear [▶ 64]</a>	Lokal	Setzt den Alarmzustand auf „Not-Raised“.
<a href="#">Confirm [▶ 65]</a>	Lokal	Bestätigt den Alarm.
<a href="#">Create [▶ 66]</a>	Lokal	Erstellt eine Alarminstanz im EventLogger.
<a href="#">CreateEx [▶ 66]</a>	Lokal	Erstellt aus eine Ereignisdefinition eine Alarminstanz im EventLogger.
<a href="#">Raise [▶ 67]</a>	Lokal	Setzt den Alarmzustand auf „Raised“.
<a href="#">SetJsonAttribute [▶ 68]</a>	Lokal	Setzt das Json-Attribut.

 **Eigenschaften**

Name	Typ	Zugriff	Definitionsort	Beschreibung
eSeverity	TcEventSeverity [▶ 113]	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert die Severity.
EventClass	GUID	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert die GUID der Ereignisklasse.
ipArguments [▶ 78]	I_TcArguments [▶ 96]	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert den Schnittstellenzeiger für die Argumente.
ipSourceInfo [▶ 78]	I_TcSourceInfo [▶ 111]	Get	Geerbt von FB_TcEventBase [▶ 72]	Als Standardverhalten wird die SourceInfo intern generiert. Sie beinhaltet dann den Symbolnamen des Funktionsbausteins, der FB_TcMessage instanziiert, als SourceName sowie die Objekt-ID der SPS-Instanz als SourceID.  Wenn die Instanz von FB_TcMessage mit dem Attribut 'hide' versteckt wird, kann intern kein Symbolname für das Standardverhalten generiert werden.
nEventId	nEventId	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert die ID des Ereignisses.
stEventEntry	TcEventEntry [▶ 113]	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert die Ereignisdefinition.
bRaised	BOOL	Get	Lokal	Liefert TRUE, wenn der Alarm im Zustand „Raised“ ist.
bActive	BOOL	Get	Lokal	Liefert TRUE, wenn der Alarm im Zustand „Raised“ oder „Wait For Confirmation“ ist.
eConfirmationState	TcEventConfirmationState [▶ 113]	Get	Lokal	Liefert den Bestätigungszustand.
nTimeCleared	ULINT	Get	Lokal	Liefert den Zeitpunkt des Clear.
nTimeConfirmed	ULINT	Get	Lokal	Liefert den Zeitpunkt des Confirm.
nTimeRaised	ULINT	Get	Lokal	Liefert den Zeitpunkt des Raise.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

**3.6.1 Clear**



Diese Methode setzt den Alarmzustand auf Not-Raised.

**Syntax**

```
METHOD Clear : HRESULT
VAR_INPUT
    nTimeStamp      : ULINT;
    bResetConfirmation : BOOL;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC).
bResetConfirmation	BOOL	Wenn TRUE und der Bestätigungszustand WaitForConfirmation ist, wird der Bestätigungszustand auf Reset gesetzt. Ansonsten wird der Bestätigungszustand nicht verändert.

 **Rückgabewert**

Name	Typ	Beschreibung
Clear	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ADS_E_INVALIDSTATE, wenn der Alarm nicht im Zustand Raised war. Liefert ansonsten ein HRESULT als Fehlercode.

**3.6.2 Confirm**



Diese Methode setzt den Bestätigungszustand von WaitForConfirmation auf Confirmed.

**Syntax**

```
METHOD Confirm : HRESULT
VAR_INPUT
    nTimeStamp: ULINT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet. > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC).

 **Rückgabewert**

Name	Typ	Beschreibung
Confirm	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ADS_E_INVALIDSTATE, wenn der Bestätigungszustand nicht WaitForConfirmation war. Liefert ansonsten ein HRESULT als Fehlercode.

### 3.6.3 Create



Diese Methode erstellt eine Alarminstanz im EventLogger.

#### Syntax

```

METHOD Create : HRESULT
    eventClass      : GUID;
    nEventId        : UDINT;
    eSeverity       : TcEventSeverity;
    bWithConfirmation : BOOL;
    ipSourceInfo    : I_TcSourceInfo;
END_VAR
    
```

#### Eingänge

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse.
nEventId	UDINT	ID des Ereignisses.
eSeverity	TcEventSeverity [▶ 113]	Severity des Ereignisses.
bWithConfirmation	BOOL	Legt fest, ob der Alarm bestätigungspflichtig ist.
ipSourceInfo	I_TcSourceInfo [▶ 111]	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ FB_TcSourceInfo [▶ 93] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.

#### Rückgabewert

Name	Typ	Beschreibung
Create	HRESULT	Liefert S_OK, wenn ein neuer Alarm erfolgreich erstellt werden konnte. Liefert ERROR_ALREADY_EXISTS, wenn der Alarm bereits existiert hat. Liefert ansonsten ein HRESULT als Fehlercode

### 3.6.4 CreateEx



Diese Methode erstellt eine Alarminstanz im EventLogger.

#### Syntax

```

METHOD CreateEx : HRESULT
VAR_INPUT
    stEventEntry : TcEventEntry;
    bWithConfirmation : BOOL;
    ipSourceInfo : I_TcSourceInfo;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
stEventEntry	TcEventEntry [ <a href="#">▶ 113</a> ]	Ereignisdefinition.
bWithConfirmation	BOOL	Legt fest, ob der Alarm bestätigungspflichtig ist.
ipSourceInfo	TcSourceInfo [ <a href="#">▶ 111</a> ]	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ FB TcSourceInfo [ <a href="#">▶ 93</a> ] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.

 **Rückgabewert**

Name	Typ	Beschreibung
CreateEx	HRESULT	Liefert S_OK, wenn ein neuer Alarm erfolgreich erstellt werden konnte. Liefert ERROR_ALREADY_EXISTS, wenn der Alarm bereits existiert hat. Liefert ansonsten ein HRESULT als Fehlercode.

### 3.6.5 Raise



Die Methode setzt den Alarmzustand auf Raised.

Wenn der Alarm bestätigungspflichtig ist, wird zusätzlich der Bestätigungszustand auf WaitForConfirmation gesetzt.

**Syntax**

```
METHOD Raise : HRESULT
VAR_INPUT
    nTimeStamp : ULINT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet. > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC).

 **Rückgabewert**

Name	Typ	Beschreibung
Raise	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ADS_E_INVALIDSTATE, wenn der Alarm bereits im Zustand Raised war. Liefert ansonsten ein HRESULT als Fehlercode.

### 3.6.6 SetJsonAttribute



Diese Methode setzt das JSON-Attribut.

#### Syntax

```

METHOD SetJsonAttribute : HRESULT
VAR_IN_OUT CONSTANT
    sJsonAttribute : STRING;
END_VAR
  
```

#### Eingänge

Name	Typ	Beschreibung
sJsonAttribute	STRING	JSON-String

#### Rückgabewert

Name	Typ	Beschreibung
SetJsonAttribute	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

## 3.7 FB\_TcArguments

**FB\_TcArguments**

Mit diesem Funktionsbaustein können Argumente eines Ereignisses definiert werden. Er implementiert dafür die I\_TcArguments-Schnittstelle.

#### Syntax

Definition:

```

FUNCTION_BLOCK FB_TcArguments IMPLEMENTS I_TcArguments
  
```

#### Schnittstellen

Typ	Beschreibung
I_TcArguments <a href="#">[► 96]</a>	Definiert das Argumenten-Handling.

 **Methoden**

Name	Definitionsort	Beschreibung
<a href="#">AddBlob [► 96]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt Binärdaten als Argument hinzu.
<a href="#">AddBool [► 97]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ BOOL hinzu.
<a href="#">AddByte [► 97]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ BYTE hinzu.
<a href="#">AddDint [► 98]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ DINT hinzu.
<a href="#">AddDWord [► 98]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ DWORD hinzu.
<a href="#">AddEventReferenceld [► 99]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt eine Referenz zu einem anderen Ereignis als Argument hinzu.
<a href="#">AddEventReferenceldG uid [► 99]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt eine Referenz zu einem anderen Ereignis als Argument hinzu.
<a href="#">AddInt [► 100]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ INT hinzu.
<a href="#">AddLInt [► 100]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ LINT hinzu.
<a href="#">AddLReal [► 100]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ LREAL hinzu.
<a href="#">AddReal [► 101]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ REAL hinzu.
<a href="#">AddSint [► 101]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ SINT hinzu.
<a href="#">AddString [► 102]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ STRING hinzu.
<a href="#">AddUDint [► 102]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ UDINT hinzu.
<a href="#">AddUInt [► 103]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ INT hinzu.
<a href="#">AddULInt [► 103]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ ULINT hinzu.
<a href="#">AddUSint [► 104]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ USINT hinzu.
<a href="#">AddWord [► 104]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ WORD hinzu.
<a href="#">AddWString [► 104]</a>	<a href="#">  TcArguments [► 96]</a>	Fügt ein Argument vom Typ WSTRING hinzu.
<a href="#">Clear [► 105]</a>	<a href="#">  TcArguments [► 96]</a>	Entfernt alle Argumente.
<a href="#">IsEmpty [► 70]</a>	Lokal	Prüft, ob Argumente hinzugefügt wurden.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
nCount	UDINT	Get	Liefert die Anzahl der übergebenen Argumente.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 3.7.1 IsEmpty



Diese Methode prüft, ob Argumente hinzugefügt wurden.

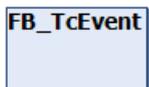
**Syntax**

```
METHOD IsEmpty : BOOL
```

 **Rückgabewert**

Name	Typ	Beschreibung
IsEmpty	BOOL	Liefert TRUE, wenn keine Argumente hinzugefügt wurden.

### 3.8 FB\_TcEvent



Dieser Funktionsbaustein stellt nur Lese-Methoden und -Eigenschaften zu einem Ereignis bereit.

**Syntax**

Definition:

```
FUNCTION_BLOCK FB_TcEvent EXTENDS FB_TcEventBase IMPLEMENTS I_TcEventBase
```

**Vererbungshierarchie**

FB\_TcEventBase [\[▶ 72\]](#)

    FB\_TcEvent

 **Schnittstellen**

Typ	Beschreibung
<a href="#">I_TcEventBase [▶ 105]</a>	Basisschnittstelle, die Methoden und Eigenschaften eines Ereignisses definiert.

 **Methoden**

Name	Definitionsort	Beschreibung
<a href="#">EqualsTo</a> [▶ 73]	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Vergleicht das Ereignis mit einer anderen Instanz.
<a href="#">EqualsToEventClass</a> [▶ 74]	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Vergleicht die Ereignisklasse des Ereignisses mit einer anderen Ereignisklasse.
<a href="#">EqualsToEventEntry</a> [▶ 74]	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Vergleicht die Ereignisdefinition des Ereignisses mit einer anderen Ereignisdefinition.
<a href="#">EqualsToEventEntryEx</a> [▶ 75]	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Vergleicht die Ereignisdefinition des Ereignisses mit einer anderen Ereignisdefinition.
<a href="#">GetJsonAttribute</a> [▶ 75]	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Liefert das Json-Attribut.
<a href="#">Release</a> [▶ 76]	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Gibt die vom EventLogger erstellte Instanz wieder frei.
<a href="#">RequestEventClassName</a> [▶ 76]	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Fragt den Namen der Ereignisklasse an.
<a href="#">RequestEventText</a> [▶ 77]	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Liefert den Text zum Ereignis.

 **Eigenschaften**

Name	Typ	Zugriff	Definitionsort	Beschreibung
eSeverity	<a href="#">TcEventSeverity</a> [▶ 113]	Get	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Liefert die Severity.
EventClass	GUID	Get	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Liefert die GUID der Ereignisklasse.
<a href="#">ipArguments</a> [▶ 78]	<a href="#">I_TcArguments</a> [▶ 96]	Get	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Liefert den Schnittstellenzeiger für die Argumente.
<a href="#">ipSourceInfo</a> [▶ 78]	<a href="#">I_TcSourceInfo</a> [▶ 111]	Get	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Als Standardverhalten wird die SourceInfo intern generiert. Sie beinhaltet dann den Symbolnamen von dem Funktionsbaustein, der <a href="#">FB_TcMessage</a> instanziiert, als SourceName sowie die Objekt-ID der SPS-Instanz als SourceID.  Wenn die Instanz von <a href="#">FB_TcMessage</a> mit dem Attribut 'hide' versteckt wird, kann intern kein Symbolname für das Standardverhalten generiert werden.
nEventId	nEventId	Get	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Liefert die ID des Ereignisses.
stEventEntry	<a href="#">TcEventEntry</a> [▶ 113]	Get	Geerbt von <a href="#">FB_TcEventBase</a> [▶ 72]	Liefert die Ereignisdefinition.
nTimestamp	ULINT	Get	Lokal	Liefert den Zeitpunkt.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

**3.9 FB\_TcEventBase**


Dieser Funktionsbaustein beinhaltet die Basisimplementierung.

**Syntax**

Definition:

```
FUNCTION_BLOCK FB_TcEventBase
```

 **Methoden**

Name	Definitionsart	Beschreibung
<a href="#">EqualsTo</a> [ <a href="#">73</a> ]	Lokal	Vergleicht das Ereignis mit einer anderen Instanz.
<a href="#">EqualsToEventClass</a> [ <a href="#">74</a> ]	Lokal	Vergleicht die Ereignisklasse des Ereignisses mit einer anderen Ereignisklasse.
<a href="#">EqualsToEventEntry</a> [ <a href="#">74</a> ]	Lokal	Vergleicht die Ereignisdefinition des Ereignisses mit einer anderen Ereignisdefinition.
<a href="#">EqualsToEventEntryEx</a> [ <a href="#">75</a> ]	Lokal	Vergleicht die Ereignisdefinition des Ereignisses mit einer anderen Ereignisdefinition.
<a href="#">GetJsonAttribute</a> [ <a href="#">75</a> ]	Lokal	Liefert das Json-Attribut.
<a href="#">Release</a> [ <a href="#">76</a> ]	Lokal	Gibt die vom EventLogger erstellte Instanz wieder frei.
<a href="#">RequestEventClassName</a> [ <a href="#">76</a> ]	Lokal	Fragt den Namen der Ereignisklasse an.
<a href="#">RequestEventText</a> [ <a href="#">77</a> ]	Lokal	Liefert den Text zum Ereignis.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
eSeverity	TcEventSeverity <a href="#"> &gt; 113</a>	Get	Liefert die Severity.
EventClass	GUID	Get	Liefert die GUID der Ereignisklasse.
ipArguments <a href="#"> &gt; 78</a>	I TcArguments <a href="#"> &gt; 96</a>	Get	Liefert den Schnittstellenzeiger für die Argumente.
ipSourceInfo <a href="#"> &gt; 78</a>	I TcSourceInfo <a href="#"> &gt; 111</a>	Get	Als Standardverhalten wird die SourceInfo intern generiert. Sie beinhaltet dann den Symbolnamen von dem Funktionsbaustein, der FB_TcMessage instanziiert, als SourceName sowie die Objekt-ID der SPS Instanz als SourceID.  Falls die Instanz von FB_TcMessage mit dem Attribut 'hide' versteckt wird, kann intern kein Symbolname für das Standardverhalten generiert werden.
nEventId	UDINT	Get	Liefert die ID des Ereignisses
nUniqueld	UDINT	Get	Liefert die Unique-ID des Ereignisses
stEventEntry	TcEventEntry <a href="#"> &gt; 113</a>	Get	Liefert die Ereignisdefinition.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

**3.9.1 EqualsTo**



Diese Methode führt einen Vergleich mit einem am Eingang angegebenen anderen Ereignis aus.

**Syntax**

```
METHOD EqualsTo : BOOL
VAR_INPUT
    ipOther : I_TcEventBase;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
ipOther	I_TcEventBase <a href="#"> &gt; 105</a>	Zu vergleichendes Ereignis

 **Rückgabewert**

Name	Typ	Beschreibung
EqualsTo	BOOL	Liefert TRUE, wenn die Ereignisse übereinstimmen.

### 3.9.2 EqualsToEventClass



Diese Methode führt einen Vergleich mit einer am Eingang angegebenen anderen Ereignisklasse aus.

**Syntax**

```
METHOD EqualsToEventClass : BOOL
VAR_INPUT
    OtherEventClass : GUID;
END_VAR
```

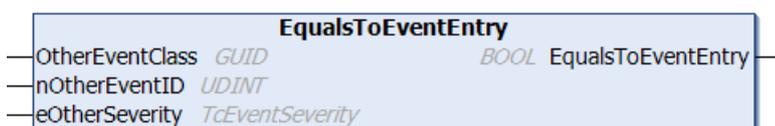
 **Eingänge**

Name	Typ	Beschreibung
OtherEventClass	GUID	Zu vergleichende Ereignisklasse.

 **Rückgabewert**

Name	Typ	Beschreibung
EqualsToEventClass	BOOL	Liefert TRUE, wenn die Ereignisklassen übereinstimmen.

### 3.9.3 EqualsToEventEntry



Diese Methode führt einen Vergleich mit einem am Eingang angegebenen anderen Ereignis aus.

**Syntax**

```
METHOD EqualsToEventEntry : BOOL
VAR_INPUT
    OtherEventClass : GUID;
    nOtherEventID : UDINT;
    eOtherSeverity : TcEventSeverity;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
OtherEventClass	GUID	Ereignisklasse des zu vergleichenden Ereignisses.
nOtherEventID	UDINT	Event-ID des zu vergleichenden Ereignisses.
eOtherSeverity	<u>TcEventSeverity</u>  ▶ 113	Event-Severity des zu vergleichenden Ereignisses.

 **Rückgabewert**

Name	Typ	Beschreibung
EqualsToEventEntry	BOOL	Liefert TRUE, wenn die Ereignisse übereinstimmen.

### 3.9.4 EqualsToEventEntryEx



Diese Methode führt einen Vergleich mit einem am Eingang angegebenen anderen Ereignis aus.

**Syntax**

```
METHOD EqualsToEventEntryEx : BOOL
VAR_INPUT
    stOther : TcEventEntry;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
stOther	TcEventEntry [ <a href="#">▶ 113</a> ]	Zu vergleichendes Ereignis.

 **Rückgabewert**

Name	Typ	Beschreibung
EqualsToEventEntryEx	BOOL	Liefert TRUE, wenn die Ereignisse übereinstimmen.

### 3.9.5 GetJsonAttribute



Diese Methode liefert das JSON-Attribut.

**Syntax**

```
METHOD GetJsonAttribute : HRESULT
VAR_INPUT
    sJsonAttribute : REFERENCE TO STRING;
    nJsonAttribute : UDINT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
sJsonAttribute	REFERENCE TO STRING	Referenz auf eine Variable vom Typ String
nJsonAttribute	UDINT	Länge der String-Variable

 Rückgabewert

Name	Typ	Beschreibung
GetJsonAttribute	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ERROR_BAD_LENGTH, wenn die Länge der Variable zu klein ist. Ansonsten wird HRESULT als Fehlercode zurückgegeben.

### 3.9.6 Release



Diese Methode gibt die vom Eventlogger erstellte Instanz wieder frei.

**Syntax**

```
METHOD Release : HRESULT
```

 Rückgabewert

Name	Typ	Beschreibung
Release	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.9.7 RequestEventClassName



Diese Methode liefert den Namen der Ereignisklasse.

**Syntax**

```
METHOD RequestEventClassName : BOOL
VAR_INPUT
    nLangId      : DINT;
    sResult      : REFERENCE TO STRING;
    nResultSize  : UDINT;
END_VAR
VAR_OUTPUT
    bError       : BOOL;
    hrErrorCode  : HRESULT;
END_VAR
```

 Eingänge

Name	Typ	Beschreibung
nLangId	DINT	Spezifiziert die Sprach-ID Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ...
sResult	REFERENCE TO STRING	Referenz auf eine Variable vom Typ String
nResultSize	UDINT	Größe der String-Variablen in Bytes

 **Rückgabewert**

Name	Typ	Beschreibung
RequestEventClassName	BOOL	Liefert TRUE, sobald die Abfrage beendet wurde. Wenn die asynchrone Abfrage noch aktiv ist, liefert der Rückgabewert FALSE. Die Methode muss so lange aufgerufen werden, bis der Rückgabewert TRUE ist.

 **Ausgänge**

Name	Typ	Beschreibung
bError	BOOL	Liefert FALSE, wenn der Methodenaufruf erfolgreich war. Liefert TRUE, wenn ein Fehler aufgetreten ist.
hrErrorCode	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Im Falle eines Fehlers wird ein Fehlercode ausgegeben.

### 3.9.8 RequestEventText



Diese Methode liefert den Ereignistext.

**Syntax**

```
METHOD RequestEventText : BOOL
VAR_INPUT
  nLangId      : DINT;
  sResult      : REFERENCE TO STRING;
  nResultSize  : UDINT;
END_VAR
VAR_OUTPUT
  bError       : BOOL;
  hrErrorCode  : HRESULT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nLangId	DINT	Spezifiziert die Sprach-ID Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ...
sResult	REFERENCE TO STRING	Referenz auf eine Variable vom Typ String
nResultSize	UDINT	Größe der String-Variable in Bytes

 **Rückgabewert**

Name	Typ	Beschreibung
RequestEventText	BOOL	Liefert TRUE, sobald die Abfrage beendet wurde. Wenn die asynchrone Abfrage noch aktiv ist, liefert der Rückgabewert FALSE. Die Methode muss so lange aufgerufen werden, bis der Rückgabewert TRUE ist.

## Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Liefert FALSE, wenn der Methodenaufruf erfolgreich war. Liefert TRUE, wenn ein Fehler aufgetreten ist.
hrErrorCode	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Im Falle eines Fehlers wird ein Fehlercode ausgegeben.

## 3.9.9 ipArguments

PROPERTY PUBLIC ipArguments : I\_TcArguments

## 3.9.10 ipSourceInfo

PROPERTY ipSourceInfo : I\_TcSourceInfo

## 3.10 FB\_TcEventLogger

**FB\_TcEventLogger**

Dieser Funktionsbaustein stellt den TwinCAT 3 EventLogger selbst dar.

### Syntax

Definition:

FUNCTION\_BLOCK FB\_TcEventLogger

### Methoden

Name	Beschreibung
<a href="#">ClearAlarms [▶ 79]</a>	Löscht aktive Alarmer.
<a href="#">ClearAllAlarms [▶ 79]</a>	Ruft Clear() für alle Alarmer im Zustand Raised auf.
<a href="#">ClearLoggedEvents [▶ 80]</a>	Löscht protokollierte Ereignisse.
<a href="#">ConfirmAlarms [▶ 81]</a>	Bestätigt Alarmer.
<a href="#">ConfirmAllAlarms [▶ 81]</a>	Ruft Confirm() für alle Alarmer mit dem Bestätigungszustand WaitForConfirmation auf.
<a href="#">ExportLoggedEvents [▶ 82]</a>	Exportiert protokollierte Ereignisse.
<a href="#">GetAlarm [▶ 83]</a>	Liefert einen Zeiger auf einen existierenden Alarm.
<a href="#">GetAlarmEx [▶ 83]</a>	Liefert einen Zeiger auf einen existierenden Alarm.
<a href="#">IsAlarmRaised [▶ 84]</a>	Fragt ab, ob ein Alarm im Zustand Raised ist.
<a href="#">IsAlarmRaisedEx [▶ 84]</a>	Fragt ab, ob ein Alarm im Zustand Raised ist.
<a href="#">SendMessage [▶ 85]</a>	Sendet eine Nachricht.
<a href="#">SendMessage2 [▶ 86]</a>	Sendet eine Nachricht.
<a href="#">SendMessageEx [▶ 87]</a>	Sendet eine Nachricht.
<a href="#">SendMessageEx2 [▶ 88]</a>	Sendet eine Nachricht.

### Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 3.10.1 ClearAlarms



Methode zum Löschen aktiver Alarme. Gibt S\_OK zurück, wenn erfolgreich.

#### Syntax

```
METHOD ClearAlarms      : HRESULT
VAR_INPUT
    nTimeStamp           : ULINT := 0;
    bResetConfirmation   : BOOL  := FALSE;
    ipFilter             : I_TcEventFilter;
END_VAR
```

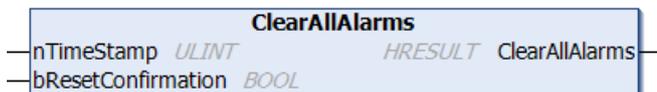
#### Eingänge

Name	Typ	Beschreibung
nTimeStamp	ULINT	0 setzen, um die aktuelle Zeit automatisch zu erhalten. Initial: 0
bResetConfirmation	BOOL	Wenn TRUE und der Bestätigungsstatus WaitForConfirmation ist, wird der Bestätigungsstatus auf Reset gesetzt. Andernfalls wird der Bestätigungsstatus nicht geändert. Initial: FALSE
ipFilter	I_TcEventFilter	Angaben, welche Alarme gelöscht werden sollen, ansonsten werden alle ausgelösten Alarme gelöscht.

#### Rückgabewerte

Name	Typ	Beschreibung
ClearAlarms	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.10.2 ClearAllAlarms



Diese Methode ruft für alle Alarme im Alarmzustand Raised deren Methode Clear() auf.

#### Syntax

```
METHOD ClearAllAlarms : HRESULT
VAR_INPUT
    nTimeStamp           : ULINT := 0;
    bResetConfirmation   : BOOL  := FALSE;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC).
bResetConfirmation	BOOL	Wenn TRUE und der Bestätigungszustand WaitForConfirmation ist, wird der Bestätigungszustand auf Reset gesetzt. Ansonsten wird der Bestätigungszustand nicht verändert.

 **Rückgabewert**

Name	Typ	Beschreibung
ClearAllAlarms	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode

### 3.10.3 ClearLoggedEvents



Async-Methode zum Löschen von protokollierten Ereignissen. Gibt TRUE zurück, wenn die asynchrone Anfrage nicht mehr belegt ist.

**Syntax**

```

METHOD ClearLoggedEvents : BOOL
VAR_INPUT
    ipClearSettings      : I_TcClearLoggedEventsSettings;
END_VAR
VAR_OUTPUT
    bError               : BOOL;
    hrErrorCode          : HRESULT;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
ipClearSettings	I_TcClearLoggedEventsSettings	Optional (Andernfalls wird der ganze Cache geleert.)

 **Rückgabewerte**

Name	Typ	Beschreibung
ClearLoggedEvents	BOOL	Liefert TRUE, sobald die Anfrage abgearbeitet ist.

 **Ausgänge**

Name	Typ	Beschreibung
bError	BOOL	Liefert TRUE, falls die Abfrage fehlerhaft ist.
hrErrorCode	HRESULT	Liefert einen Fehlercode, falls die Abfrage fehlerhaft ist.

**Beispiel für die Verwendung der Methode**

Diese Methode ist asynchron, d. h., sie benötigt mehrere Zyklen, um den Rückgabewert zu liefern. Folgende Verwendung ist vorgesehen:

```
IF bClearLoggedEvents THEN
    bClearLoggedEvents:= NOT fbLogger.ClearLoggedEvents(0);
END_IF
```

**3.10.4 ConfirmAlarms**



Diese Methode bestätigt Alarmer.

**Syntax**

```
METHOD ConfirmAlarms : HRESULT
VAR_INPUT
    nTimeStamp      : ULINT := 0;
    ipFilter        : I_TcEventFilter;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
nTimeStamp	ULINT	0 setzen, um die aktuelle Zeit automatisch zu erhalten. Initial: 0
ipFilter	I_TcEventFilter	Festlegen, welche Alarmer bestätigt werden sollen, ansonsten werden alle Alarmer mit dem Bestätigungsstatus WaitForConfirmation bestätigt.

**Rückgabewerte**

Name	Typ	Beschreibung
ConfirmAlarms	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

**3.10.5 ConfirmAllAlarms**



Diese Methode ruft für alle Alarmer mit dem Bestätigungszustand WaitForConfirmation deren Methode Confirm() auf.

**Syntax**

```
METHOD ConfirmAllAlarms : HRESULT
VAR_INPUT
    nTimeStamp : ULINT := 0;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC).

 **Rückgabewert**

Name	Typ	Beschreibung
ConfirmAllAlarms	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.10.6 ExportLoggedEvents



Diese Methode exportiert protokollierte Ereignisse asynchron. Gibt TRUE zurück, wenn die asynchrone Abarbeitung abgeschlossen ist.

**Syntax**

```

METHOD ExportLoggedEvents : BOOL
VAR_IN_OUT CONSTANT
  sFileName          : STRING;
END_VAR
VAR_INPUT
  ipExportSettings   : I_TcEventExportSettings;
END_VAR
VAR_OUTPUT
  bError             : BOOL;
  hrErrorCode        : HRESULT;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
ipExportSettings	I_TcEventExportSettings	Angeben, welche Ereignisse exportiert werden sollen, ansonsten werden alle Ereignisse exportiert. Hierfür kann eine Instanz von <a href="#">FB_TcEventCsvExportSettings</a>   411 zugewiesen werden.

 **Ein-/Ausgänge**

Name	Typ	Beschreibung
sFileName	STRING	Name der Zielfeile

 **Rückgabewerte**

Name	Typ	Beschreibung
ExportLoggedEvents	BOOL	TRUE, wenn die Abarbeitung abgeschlossen ist.

 **Ausgänge**

Name	Typ	Beschreibung
bError	BOOL	TRUE, sobald ein Fehler eintritt.
hrErrorCode	HRESULT	Gibt die Fehlerinformation aus, wenn bError TRUE ist.

**Beispiel für die Verwendung der Methode**

Diese Methode ist asynchron, d. h., sie benötigt mehrere Zyklen, um den Rückgabewert zu liefern. Folgende Verwendung ist vorgesehen:

```
IF bExportLoggedEvents THEN
  bExportLoggedEvents:= NOT fbLogger.ExportLoggedEvents(...);
END_IF
```

**3.10.7 GetAlarm**



Diese Methode liefert einen Schnittstellenzeiger auf eine existierende Instanz.

**Syntax**

```
METHOD GetAlarm : HRESULT
VAR_INPUT
  eventClass : GUID;
  nEventId : UDINT;
  ipSourceInfo : I_TcSourceInfo := 0;
  fbAlarm : REFERENCE TO FB_TcAlarm;
END_VAR
```

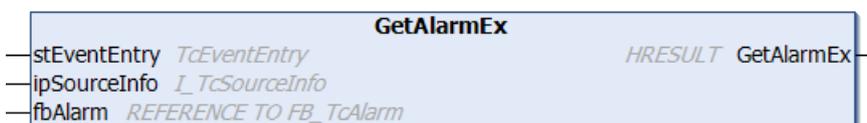
 **Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse.
nEventId	UDINT	ID des Ereignisses
ipSourceInfo	I_TcSourceInfo <a href="#">[▶ 111]</a>	Zeiger auf eine ITcSourceInfo-Schnittstelle.
fbAlarm	REFERENCE TO FB_TcAlarm <a href="#">[▶ 62]</a>	Zeiger auf einen Alarm.

 **Rückgabewert**

Name	Typ	Beschreibung
GetAlarm	HRESULT	Liefert ADS_E_NOTFOUND, wenn keine Instanz gefunden wurde. Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

**3.10.8 GetAlarmEx**



Diese Methode liefert einen Schnittstellenzeiger auf eine existierende Instanz.

**Syntax**

```
METHOD GetAlarmEx : HRESULT
VAR_INPUT
    stEventEntry : TcEventEntry;
    ipSourceInfo : I_TcSourceInfo := 0; // optional
    fbAlarm      : REFERENCE TO FB_TcAlarm;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
stEventEntry	TcEventEntry [▶ 113]	Ereignisdefinition
ipSourceInfo	I_TcSourceInfo [▶ 111]	Zeiger auf eine ITcSourceInfo-Schnittstelle
fbAlarm	REFERENCE TO FB_TcAlarm [▶ 62]	Zeiger auf einen Alarm

 **Rückgabewert**

Name	Typ	Beschreibung
GetAlarmEx	HRESULT	Liefert ADS_E_NOTFOUND, wenn keine Instanz gefunden wurde. Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

**3.10.9 IsAlarmRaised**



Diese Methode fragt ab, ob ein Alarm im Zustand Raised ist.

**Syntax**

```
METHOD IsAlarmRaised : BOOL
VAR_INPUT
    eventClass : GUID;
    nEventId   : UDINT;
    ipSourceInfo : I_TcSourceInfo := 0;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse.
nEventId	UDINT	ID des Ereignisses.
ipSourceInfo	I_TcSourceInfo [▶ 111]	Zeiger auf eine ITcSourceInfo-Schnittstelle.

 **Rückgabewert**

Name	Typ	Beschreibung
IsAlarmRaised	BOOL	Liefert TRUE, wenn der Alarm im Zustand Raised ist.

**3.10.10 IsAlarmRaisedEx**



Diese Methode fragt ab, ob ein Alarm im Zustand Raised ist.

**Syntax**

```
METHOD IsAlarmRaisedEx : BOOL
VAR_INPUT
    stEventEntry : TcEventEntry;
    ipSourceInfo : I_TcSourceInfo := 0;
END_VAR
```

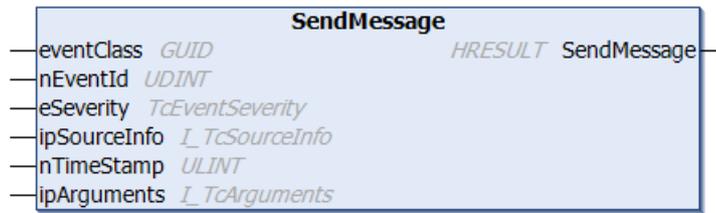
 **Eingänge**

Name	Typ	Beschreibung
stEventEntry	UDINT	Ereignisdefinition.
ipSourceInfo	I_TcSourceInfo [ <a href="#">▶ 111</a> ]	Zeiger auf eine ITcSourceInfo-Schnittstelle.

 **Rückgabewert**

Name	Typ	Beschreibung
IsAlarmRaisedEx	BOOL	Liefert TRUE, wenn der Alarm im Zustand Raised ist.

**3.10.11 SendMessage**



Diese Methode sendet eine Nachricht.

**Syntax**

```
METHOD SendMessage : HRESULT
VAR_INPUT
    eventClass : GUID;
    nEventId : UDINT;
    eSeverity : TcEventSeverity;
    ipSourceInfo : I_TcSourceInfo := 0;
    nTimeStamp : ULINT := 0;
    ipArguments : I_TcArguments := 0;
END_VAR
```

### 📁 Eingänge

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse.
nEventId	UDINT	ID des Ereignisses.
eSeverity	<a href="#">TcEventSeverity</a> [ <a href="#">▶ 113</a> ]	Severity des Ereignisses.
ipSourceInfo	<a href="#">I_TcSourceInfo</a> [ <a href="#">▶ 111</a> ]	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcSourceInfo</a> [ <a href="#">▶ 93</a> ] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet. > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC).
ipArguments	<a href="#">I_TcArguments</a> [ <a href="#">▶ 96</a> ]	Schnittstellenzeiger auf Argumente des Ereignisses. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcArguments</a> [ <a href="#">▶ 68</a> ] angegeben werden.

### 📁 Rückgabewert

Name	Typ	Beschreibung
SendMessage	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

## 3.10.12 SendMessage2



Diese Methode sendet eine Nachricht.

### Syntax

```

METHOD SendMessage2 : HRESULT
VAR_INPUT
  eventClass      : GUID;
  nEventId       : UDINT;
  eSeverity      : TcEventSeverity;
  ipSourceInfo   : I_TcSourceInfo := 0;
  nTimeStamp     : ULINT := 0;
  ipArguments    : I_TcArguments := 0;
END_VAR
VAR_IN_OUT CONSTANT
  sJsonAttribute : STRING;
END_VAR

```

 **Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse
nEventId	UDINT	ID des Ereignisses
eSeverity	TcEventSeverity	Severity des Ereignisses
ipSourceInfo	I_TcSourceInfo	Schnittstellenzeiger auf die Quellinformationen.  Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcSourceInfo</a> [► 93] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet.  > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC)
ipArguments	I_TcArguments	Schnittstellenzeiger auf Argumente des Ereignisses.  Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcArguments</a> [► 68] angegeben werden.
sJsonAttribute	STRING	String als Json Repräsentation, der zusätzlich zum Ereignistext übertragen wird.

 **Rückgabewerte**

Name	Typ	Beschreibung
SendMessage2	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.10.13 SendMessageEx

```

SendMessageEx
—stEventEntry TcEventEntry HRESULT SendMessageEx
—ipSourceInfo I_TcSourceInfo
—nTimeStamp ULINT
—ipArguments I_TcArguments
    
```

Diese Methode sendet eine Nachricht.

**Syntax**

```

METHOD SendMessageEx : HRESULT
VAR_INPUT
    stEventEntry : TcEventEntry;
    ipSourceInfo : I_TcSourceInfo := 0;
    nTimeStamp   : ULINT := 0;
    ipArguments  : I_TcArguments := 0;
END_VAR
    
```

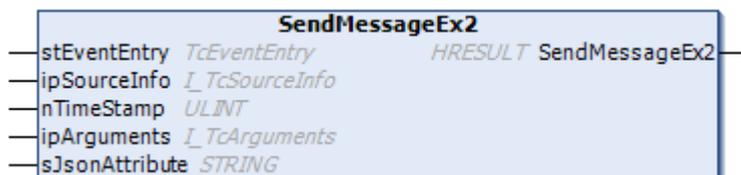
### 🚩 Eingänge

Name	Typ	Beschreibung
stEventEntry	<a href="#">TcEventEntry</a> [▶ 113]	Ereignisdefinition.
ipSourceInfo	<a href="#">I_TcSourceInfo</a> [▶ 111]	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB TcSourceInfo</a> [▶ 93] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC).
ipArguments	<a href="#">I_TcArguments</a> [▶ 96]	Schnittstellenzeiger auf Argumente des Ereignisses. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB TcArguments</a> [▶ 68] angegeben werden.

### 🚩 Rückgabewert

Name	Typ	Beschreibung
SendMessageEx	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

## 3.10.14 SendMessageEx2



Diese Methode sendet eine Nachricht.

### Syntax

```
METHOD SendMessageEx2 : HRESULT
VAR_INPUT
    stEventEntry      : TcEventEntry;
    ipSourceInfo      : I_TcSourceInfo := 0;
    nTimeStamp        : ULINT := 0;
    ipArguments       : I_TcArguments := 0;
END_VAR
VAR_IN_OUT CONSTANT
    sJsonAttribute    : STRING;
END_VAR
```

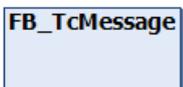
 **Eingänge**

Name	Typ	Beschreibung
stEventEntry	TcEventEntry	Ereignisdefinition
ipSourceInfo	I_TcSourceInfo	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcSourceInfo [▶ 93]</a> angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet. > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC)
ipArguments	I_TcArguments	Schnittstellenzeiger auf Argumente des Ereignisses. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcArguments [▶ 68]</a> angegeben werden.
sJsonAttribute	STRING	String als Json Repräsentation, der zusätzlich zum Ereignistext übertragen wird.

 **Rückgabewerte**

Name	Typ	Beschreibung
SendMessageEx2	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.11 FB\_TcMessage



Dieser Funktionsbaustein repräsentiert eine Nachricht vom TwinCAT 3 EventLogger.

**Syntax**

Definition:

```
FUNCTION_BLOCK FB_TcMessage EXTENDS FB_TcEventBase IMPLEMENTS I_TcMessage
```

**Vererbungshierarchie**

[FB\\_TcEventBase \[▶ 72\]](#)

FB\_TcMessage

 **Schnittstellen**

Typ	Beschreibung
<a href="#">I TcMessage</a> [ <a href="#">▶ 110</a> ]	Stellt Methoden und Eigenschaften für das Nachrichten-Handling bereit.

 **Methoden**

Name	Definitionsort	Beschreibung
<a href="#">EqualsTo</a> [ <a href="#">▶ 73</a> ]	Geerbt von <a href="#">FB_TcEventBase</a> [ <a href="#">▶ 72</a> ]	Vergleicht das Ereignis mit einer anderen Instanz.
<a href="#">EqualsToEventClass</a> [ <a href="#">▶ 74</a> ]	Geerbt von <a href="#">FB_TcEventBase</a> [ <a href="#">▶ 72</a> ]	Vergleicht die Ereignisklasse des Ereignisses mit einer anderen Ereignisklasse.
<a href="#">EqualsToEventEntry</a> [ <a href="#">▶ 74</a> ]	Geerbt von <a href="#">FB_TcEventBase</a> [ <a href="#">▶ 72</a> ]	Vergleicht die Ereignisdefinition des Ereignisses mit einer anderen Ereignisdefinition.
<a href="#">EqualsToEventEntryEx</a> [ <a href="#">▶ 75</a> ]	Geerbt von <a href="#">FB_TcEventBase</a> [ <a href="#">▶ 72</a> ]	Vergleicht die Ereignisdefinition des Ereignisses mit einer anderen Ereignisdefinition.
<a href="#">GetJsonAttribute</a> [ <a href="#">▶ 75</a> ]	Geerbt von <a href="#">FB_TcEventBase</a> [ <a href="#">▶ 72</a> ]	Liefert das Json-Attribut.
<a href="#">Release</a> [ <a href="#">▶ 76</a> ]	Geerbt von <a href="#">FB_TcEventBase</a> [ <a href="#">▶ 72</a> ]	Gibt die vom EventLogger erstellte Instanz wieder frei.
<a href="#">RequestEventClassName</a> [ <a href="#">▶ 76</a> ]	Geerbt von <a href="#">FB_TcEventBase</a> [ <a href="#">▶ 72</a> ]	Fragt den Namen der Ereignisklasse an.
<a href="#">RequestEventText</a> [ <a href="#">▶ 77</a> ]	Geerbt von <a href="#">FB_TcEventBase</a> [ <a href="#">▶ 72</a> ]	Liefert den Text zum Ereignis.
<a href="#">Create</a> [ <a href="#">▶ 91</a> ]	Lokal	Erstellt eine Nachrichtinstanz im EventLogger.
<a href="#">CreateEx</a> [ <a href="#">▶ 92</a> ]	Lokal	Erstellt aus eine Ereignisdefinition eine Nachrichtinstanz im EventLogger.
<a href="#">SetJsonAttribute</a> [ <a href="#">▶ 92</a> ]	Lokal	Setzt das Json-Attribut.
<a href="#">Send</a> [ <a href="#">▶ 110</a> ]	<a href="#">I TcMessage</a> [ <a href="#">▶ 110</a> ]	Sendet eine Nachricht.

 **Eigenschaften**

Name	Typ	Zugriff	Definitionsart	Beschreibung
eSeverity	TcEventSeverity [▶ 113]	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert die Severity.
EventClass	GUID	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert die GUID der Ereignisklasse.
ipArguments [▶ 78]	I_TcArguments [▶ 96]	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert den Schnittstellenzeiger für die Argumente.
ipSourceInfo [▶ 78]	I_TcSourceInfo [▶ 111]	Get	Geerbt von FB_TcEventBase [▶ 72]	Als Standardverhalten wird die SourceInfo intern generiert. Sie beinhaltet dann den Symbolnamen von dem Funktionsbaustein, der FB_TcMessage instanziiert, als SourceName sowie die Objekt-ID der SPS-Instanz als SourceID.  Wenn die Instanz von FB_TcMessage mit dem Attribut 'hide' versteckt wird, kann intern kein Symbolname für das Standardverhalten generiert werden.
nEventId	nEventId	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert die ID des Ereignisses.
stEventEntry	TcEventEntry [▶ 113]	Get	Geerbt von FB_TcEventBase [▶ 72]	Liefert die Ereignisdefinition.
nTimeSent	ULINT	Get	Lokal	Liefer den Zeitpunkt des Send.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

**3.11.1 Create**



Diese Methode erstellt eine Nachrichtinstanz im EventLogger.

**Syntax**

```

METHOD Create : HRESULT
VAR_INPUT
    eventClass    : GUID;
    nEventId     : UDINT;
    eSeverity     : TcEventSeverity;
    ipSourceInfo : I_TcSourceInfo := 0;
END_VAR
    
```

 **Eingänge**

Name	Typ	Beschreibung
eventClass	GUID	GUID der Ereignisklasse.
nEventId	UDINT	ID des Ereignisses.
eSeverity	<a href="#">TcEventSeverity</a> [▶ 113]	Definiert die Severity.
ipSourceInfo	<a href="#">I_TcSourceInfo</a> [▶ 111]	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcSourceInfo</a> [▶ 93] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.

 **Rückgabewert**

Name	Typ	Beschreibung
Create	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.11.2 CreateEx



Diese Methode erstellt aus eine Ereignisdefinition eine Nachrichtinstanz im EventLogger.

**Syntax**

```
METHOD PUBLIC CreateEx : HRESULT
VAR_INPUT
    stEventEntry : TcEventEntry;
    ipSourceInfo : I_TcSourceInfo := 0;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
stEventEntry	<a href="#">TcEventEntry</a> [▶ 113]	Ereignisdefinition.
ipSourceInfo	<a href="#">I_TcSourceInfo</a> [▶ 111]	Schnittstellenzeiger auf die Quellinformationen. Diese Angabe ist optional. Hier kann eine Instanz vom Typ <a href="#">FB_TcSourceInfo</a> [▶ 93] angegeben werden. Wird nichts (NULL) übergeben, wird eine Standardquellinformation generiert.

 **Rückgabewert**

Name	Typ	Beschreibung
CreateEx	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.11.3 SetJsonAttribute



Diese Methode setzt das JSON-Attribut.

**Syntax**

```
METHOD SetJsonAttribute : HRESULT
VAR_IN_OUT CONSTANT
    sJsonAttribute : STRING;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
sJsonAttribute	STRING	JSON-String

 **Rückgabewert**

Name	Typ	Beschreibung
SetJsonAttribute	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode.

### 3.12 FB\_TcSourceInfo



Mit diesem Funktionsbaustein kann die Quellinformation eines Ereignisses definiert werden.

**Syntax**

Definition:

```
FUNCTION_BLOCK FB_TcSourceInfo IMPLEMENTS I_TcSourceInfo
```

 **Schnittstellen**

Typ	Beschreibung
I_TcSourceInfo <a href="#">[▶ 111]</a>	Stellt Lese-Methoden und -Eigenschaften einer Quellinformation bereit.

 **Methoden**

Name	Definitionsort	Beschreibung
<a href="#">Clear [▶ 94]</a>	Lokal	Setzt die Quellinformation zurück.
<a href="#">ExtendName [▶ 94]</a>	Lokal	Fügt den übergebenen String an den Namen an.
<a href="#">ResetToDefault [▶ 95]</a>	Lokal	Setzt die Eigenschaften auf Standardwerte. sName wird mit dem Symbolnamen des instanzierenden Funktionsbausteins initialisiert. nId wird mit der Objekt-ID der SPS Instanz initialisiert. Wenn die Instanz von FB_TcSourceInfo mit dem Attribut 'hide' versteckt wird, kann intern kein Symbolname für das Standardverhalten generiert werden.
<a href="#">EqualsTo [▶ 111]</a>	<a href="#">I_TcSourceInfo [▶ 111]</a>	Vergleicht eine Instanz mit einer anderen Instanz.

 **Eigenschaften**

Name	Typ	Zugriff	Definitionsort	Beschreibung
guid	GUID	Get	<a href="#">  TcSourceInfo  &gt;_111</a>	Liefert die GUID der Quelleinfo zurück.
guid	GUID	SET	Lokal	Setzt die GUID als Quelleinfo.
nId	UDINT	Get	<a href="#">  TcSourceInfo  &gt;_111</a>	Liefert die ID der Quelleinfo.
nId	UDINT	SET	Lokal	Setzt die ID der Quelleinfo.
sName	STRING(ParameterList.cSourceNameSize-1)	Get	<a href="#">  TcSourceInfo  &gt;_111</a>	Liefert den Namen der Quelleinfo.
sName	STRING(ParameterList.cSourceNameSize-1)	SET	Lokal	Setzt den Namen der Quelleinfo

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

**3.12.1 Clear**

Diese Methode setzt die Quellinformation zurück.

**Syntax**

```
METHOD Clear
```

**3.12.2 ExtendName**

Diese Methode erweitert den Namen.

**Syntax**

```
METHOD ExtendName : BOOL
VAR_INPUT
    sExtension : STRING(255);
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
sExtension	STRING(255)	Text der rechts angehängt werden soll.

**👉 Rückgabewert**

Name	Typ	Beschreibung
ExtendName	BOOL	Liefert TRUE, wenn die Verkettung erfolgreich war. Liefert FALSE, wenn die resultierende Zeichenfolge länger ist als die Ausgabe-Zeichenfolge und nicht in den gegebenen Ausgangspuffer passt. Der Speicherbedarf der resultierenden Zeichenfolge ist dann größer als der der Ausgabe-Zeichenfolge. Die Zeichenfolge wird dann abgeschnitten.

### 3.12.3 ResetToDefault

**ResetToDefault**

Diese Methode setzt die Quellinformation auf Standardwerte.

**Standardwerte:**

sName wird mit dem Symbolnamen des instanziiierenden Funktionsbausteins initialisiert.

nId wird mit der Objekt-ID der SPS-Instanz initialisiert.

Wenn die Instanz von FB\_TcSourceInfo mit dem Attribut 'hide' versteckt wird, kann intern kein Symbolname für das Standardverhalten generiert werden.

**Syntax**

```
METHOD ResetToDefault
```

## 4 Schnittstellen

### 4.1 I\_TcArguments

Diese Schnittstelle definiert Methoden für das Argumenten-Handling.

#### Vererbungshierarchie

\_\_SYSTEM.IQueryInterface

I\_TcArguments

#### Methoden

Name	Beschreibung
<a href="#">AddBlob [▶ 96]</a>	Fügt Binärdaten als Argument hinzu.
<a href="#">AddBool [▶ 97]</a>	Fügt ein Argument vom Typ BOOL hinzu.
<a href="#">AddByte [▶ 97]</a>	Fügt ein Argument vom Typ BYTE hinzu.
<a href="#">AddDint [▶ 98]</a>	Fügt ein Argument vom Typ DINT hinzu.
<a href="#">AddDWord [▶ 98]</a>	Fügt ein Argument vom Typ DWORD hinzu.
<a href="#">AddEventReferenceld [▶ 99]</a>	Fügt eine Referenz zu einem anderen Ereignis als Argument hinzu.
<a href="#">AddEventReferenceldGuid [▶ 99]</a>	Fügt eine Referenz zu einem anderen Ereignis als Argument hinzu.
<a href="#">AddInt [▶ 100]</a>	Fügt ein Argument vom Typ INT hinzu.
<a href="#">AddLInt [▶ 100]</a>	Fügt ein Argument vom Typ LINT hinzu.
<a href="#">AddLReal [▶ 100]</a>	Fügt ein Argument vom Typ LREAL hinzu.
<a href="#">AddReal [▶ 101]</a>	Fügt ein Argument vom Typ REAL hinzu.
<a href="#">AddSint [▶ 101]</a>	Fügt ein Argument vom Typ SINT hinzu.
<a href="#">AddString [▶ 102]</a>	Fügt ein Argument vom Typ STRING hinzu.
<a href="#">AddUDint [▶ 102]</a>	Fügt ein Argument vom Typ UDINT hinzu.
<a href="#">AddUInt [▶ 103]</a>	Fügt ein Argument vom Typ INT hinzu.
<a href="#">AddULInt [▶ 103]</a>	Fügt ein Argument vom Typ ULINT hinzu.
<a href="#">AddUSint [▶ 104]</a>	Fügt ein Argument vom Typ USINT hinzu.
<a href="#">AddWord [▶ 104]</a>	Fügt ein Argument vom Typ WORD hinzu.
<a href="#">AddWString [▶ 104]</a>	Fügt ein Argument vom Typ WSTRING hinzu.
<a href="#">Clear [▶ 105]</a>	Entfernt alle Argumente.

#### Eigenschaften

Name	Typ	Zugriff	Beschreibung
nCount	UDINT	Get	Liefert die Anzahl der übergebenen Argumente.

#### 4.1.1 AddBlob

AddBlob	
pData <i>PVOID</i>	I_TcArguments AddBlob
cbData <i>UINT</i>	

Diese Methode fügt Binärdaten als Argument hinzu.

**Syntax**

```
METHOD AddBlob : I_TcArguments
VAR_INPUT
    pData : PVOID;
    cbData : UINT;
END_VAR
```

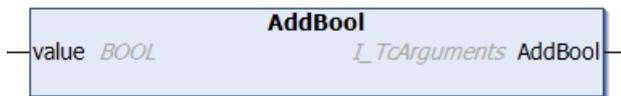
**Eingänge**

Name	Typ	Beschreibung
pData	PVOID	Zeiger auf das erste Byte von dem Binärdatum.
cbData	UINT	Länge des Binärdatums in Bytes.

**Rückgabewert**

Name	Typ	Beschreibung
AddBlob	I_TcArguments [ <a href="#">▶ 96</a> ]	Liefert den I_TcArgument-Zeiger wieder zurück.

**4.1.2 AddBool**



Diese Methode fügt ein Argument vom Typ BOOL hinzu.

**Syntax**

```
METHOD AddBool : I_TcArguments
VAR_INPUT
    value : BOOL;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
value	BOOL	Wert, der hinzugefügt werden soll.

**Rückgabewert**

Name	Typ	Beschreibung
AddBool	I_TcArguments [ <a href="#">▶ 96</a> ]	Liefert den I_TcArgument-Zeiger wieder zurück.

**4.1.3 AddByte**



Diese Methode fügt ein Argument vom Typ BYTE hinzu.

**Syntax**

```
METHOD AddByte : I_TcArguments
VAR_INPUT
    value : BYTE;
END_VAR
```

### 📌 Eingänge

Name	Typ	Beschreibung
value	BYTE	Wert, der hinzugefügt werden soll.

### 📌 Rückgabewert

Name	Typ	Beschreibung
AddByte	I_TcArguments [▶ 96]	Liefert den I_TcArgument-Zeiger wieder zurück.

## 4.1.4 AddDint



Diese Methode fügt ein Argument vom Typ DINT hinzu.

### Syntax

```

METHOD AddDINT : I_TcArguments
VAR_INPUT
    value : DINT;
END_VAR
  
```

### 📌 Eingänge

Name	Typ	Beschreibung
value	DINT	Wert, der hinzugefügt werden soll.

### 📌 Rückgabewert

Name	Typ	Beschreibung
AddDINT	I_TcArguments [▶ 96]	Liefert den I_TcArgument-Zeiger wieder zurück.

## 4.1.5 AddDWord



Diese Methode fügt ein Argument vom Typ DWORD hinzu.

### Syntax

```

METHOD AddDWord : I_TcArguments
VAR_INPUT
    value : DWORD;
END_VAR
  
```

### 📌 Eingänge

Name	Typ	Beschreibung
value	DWORD	Wert, der hinzugefügt werden soll.

 Rückgabewert

Name	Typ	Beschreibung
AddDWord	I_TcArguments [▶ 96]	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.6 AddEventReferenceId



Diese Methode fügt eine Referenz zu einem anderen Ereignis als Argument hinzu.

**Syntax**

```
METHOD AddEventReferenceId : I_TcArguments
VAR_INPUT
    nEventId : UDINT;
END_VAR
```

 Eingänge

Name	Typ	Beschreibung
nEventId	UDINT	ID des Ereignisses.

 Rückgabewert

Name	Typ	Beschreibung
AddEventReferenceId	I_TcArguments [▶ 96]	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.7 AddEventReferenceIdGuid



Diese Methode fügt eine Referenz zu einem anderen Ereignis als Argument hinzu.

**Syntax**

```
METHOD AddEventReferenceIdGuid : I_TcArguments
VAR_INPUT
    nEventId : UDINT;
    EventClass : GUID;
END_VAR
```

 Eingänge

Name	Typ	Beschreibung
nEventId	UDINT	ID des Ereignisses.
EventClass	GUID	GUID der Ereignisklasse.

 Rückgabewert

Name	Typ	Beschreibung
AddEventReferenceIdGuid	I_TcArguments [▶ 96]	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.8 AddInt



Diese Methode fügt ein Argument vom Typ INT hinzu.

#### Syntax

```
METHOD AddINT : I_TcArguments
VAR_INPUT
    value : INT;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
value	INT	Wert, der hinzugefügt werden soll.

#### Rückgabewert

Name	Typ	Beschreibung
AddInt	I_TcArguments <a href="#"> &gt; 96</a>	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.9 AddLInt



Diese Methode fügt ein Argument vom Typ LINT hinzu.

#### Syntax

```
METHOD AddLInt : I_TcArguments
VAR_INPUT
    value : LINT;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
value	LINT	Wert, der hinzugefügt werden soll.

#### Rückgabewert

Name	Typ	Beschreibung
AddLInt	I_TcArguments <a href="#"> &gt; 96</a>	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.10 AddLReal



Diese Methode fügt ein Argument vom Typ LREAL hinzu.

**Syntax**

```
METHOD AddLReal : I_TcArguments
VAR_INPUT
    value : LREAL;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
value	LREAL	Wert, der hinzugefügt werden soll.

 **Rückgabewert**

Name	Typ	Beschreibung
AddLReal	I_TcArguments <a href="#"> ▶ 96 </a>	Liefert den I_TcArgument-Zeiger wieder zurück.

**4.1.11 AddReal**



Diese Methode fügt ein Argument vom Typ REAL hinzu.

**Syntax**

```
METHOD AddReal : I_TcArguments
VAR_INPUT
    value : REAL;
END_VAR
```

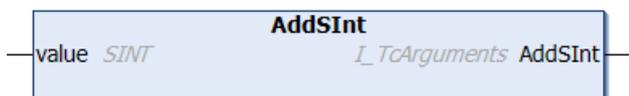
 **Eingänge**

Name	Typ	Beschreibung
value	REAL	Wert, der hinzugefügt werden soll.

 **Rückgabewert**

Name	Typ	Beschreibung
AddReal	I_TcArguments <a href="#"> ▶ 96 </a>	Liefert den I_TcArgument-Zeiger wieder zurück.

**4.1.12 AddSInt**



Diese Methode fügt ein Argument vom Typ SINT hinzu.

**Syntax**

```
METHOD AddSInt : I_TcArguments
VAR_INPUT
    value : SInt;
END_VAR
```

### Eingänge

Name	Typ	Beschreibung
value	SINT	Wert, der hinzugefügt werden soll.

### Rückgabewert

Name	Typ	Beschreibung
AddSInt	<a href="#">I_TcArguments</a>   <a href="#">96</a>	Liefert den <a href="#">I_TcArgument</a> -Zeiger wieder zurück.

## 4.1.13 AddString



Diese Methode fügt ein Argument vom Typ STRING hinzu.

### Syntax

```
METHOD AddString : I_TcArguments
VAR_IN_OUT CONSTANT
    value : STRING;
END_VAR
```

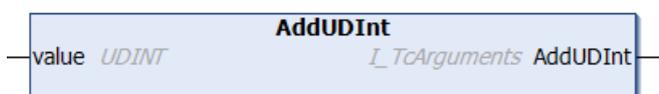
### Eingänge

Name	Typ	Beschreibung
value	STRING	Wert, der hinzugefügt werden soll.

### Rückgabewert

Name	Typ	Beschreibung
AddString	<a href="#">I_TcArguments</a>   <a href="#">96</a>	Liefert den <a href="#">I_TcArgument</a> -Zeiger wieder zurück.

## 4.1.14 AddUDInt



Diese Methode fügt ein Argument vom Typ UDINT hinzu.

### Syntax

```
METHOD AddUDInt : I_TcArguments
VAR_INPUT
    value : UDINT;
END_VAR
```

### Eingänge

Name	Typ	Beschreibung
value	UDINT	Wert, der hinzugefügt werden soll.

 Rückgabewert

Name	Typ	Beschreibung
AddUDInt	I_TcArguments <a href="#">▶ 96</a>	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.15 AddUInt



Diese Methode fügt ein Argument vom Typ INT hinzu.

**Syntax**

```
METHOD AddUInt : I_TcArguments
VAR_INPUT
    value : UINT;
END_VAR
```

 Eingänge

Name	Typ	Beschreibung
value	UINT	Wert, der hinzugefügt werden soll.

 Rückgabewert

Name	Typ	Beschreibung
AddUInt	I_TcArguments <a href="#">▶ 96</a>	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.16 AddULInt



Diese Methode fügt ein Argument vom Typ ULINT hinzu.

**Syntax**

```
METHOD AddULInt : I_TcArguments
VAR_INPUT
    value : ULINT;
END_VAR
```

 Eingänge

Name	Typ	Beschreibung
value	ULINT	Wert, der hinzugefügt werden soll.

 Rückgabewert

Name	Typ	Beschreibung
AddULInt	I_TcArguments <a href="#">▶ 96</a>	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.17 AddUSInt



Diese Methode fügt ein Argument vom Typ USINT hinzu.

#### Syntax

```
METHOD AddUSInt : I_TcArguments
VAR_INPUT
    value : USINT;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
value	USINT	Wert, der hinzugefügt werden soll.

#### Rückgabewert

Name	Typ	Beschreibung
AddUSInt	I_TcArguments [ <a href="#">▶ 96</a> ]	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.18 AddWord



Diese Methode fügt ein Argument vom Typ WORD hinzu.

#### Syntax

```
METHOD AddWord : I_TcArguments
VAR_INPUT
    value : WORD;
END_VAR
```

#### Eingänge

Name	Typ	Beschreibung
value	WORD	Wert, der hinzugefügt werden soll.

#### Rückgabewert

Name	Typ	Beschreibung
AddWord	I_TcArguments [ <a href="#">▶ 96</a> ]	Liefert den I_TcArgument-Zeiger wieder zurück.

### 4.1.19 AddWString



Diese Methode fügt ein Argument vom Typ WSTRING hinzu.

**Syntax**

```
METHOD AddWString : I_TcArguments
VAR_IN_OUT CONSTANT
    value : WSTRING;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
value	WSTRING	Wert, der hinzugefügt werden soll.

 **Rückgabewert**

Name	Typ	Beschreibung
AddWString	I_TcArguments <a href="#"> &gt; 96</a>	Liefert den I_TcArgument-Zeiger wieder zurück.

**4.1.20 Clear**



Diese Methode entfernt alle Argumente.

**Syntax**

```
METHOD Clear : I_TcArguments
```

 **Rückgabewert**

Name	Typ	Beschreibung
Clear	I_TcArguments <a href="#"> &gt; 96</a>	Liefert den I_TcArgument-Zeiger wieder zurück.

**4.2 I\_TcEventBase**

In dieser Basisschnittstelle sind Methoden und Eigenschaften eines Ereignisses definiert.

 **Methoden**

Name	Beschreibung
<a href="#">EqualsTo  &gt; 106</a>	Vergleicht das Ereignis mit einer anderen Instanz.
<a href="#">EqualsToEventClass  &gt; 106</a>	Vergleicht die Ereignisklasse des Ereignisses mit einer anderen Ereignisklasse.
<a href="#">EqualsToEventEntryEx  &gt; 107</a>	Vergleicht die Ereignisdefinition des Ereignisses mit einer anderen Ereignisdefinition.
<a href="#">GetJsonAttribute  &gt; 108</a>	Liefert das Json-Attribut.
<a href="#">RequestEventClassName  &gt; 108</a>	Fragt den Namen der Ereignisklasse an.
<a href="#">RequestEventText  &gt; 109</a>	Liefert den Text zum Ereignis.

## Eigenschaften

Name	Typ	Zugriff	Beschreibung
eSeverity	TcEventSeverity [▶ 113]	Get	Liefert die Severity.
EventClass	GUID	Get	Liefert die GUID der Ereignisklasse.
ipSourceInfo	I_TcSourceInfo [▶ 111]	Get	Liefert einen Zeiger auf die Quelldefinition.
nEventId	UDINT	Get	Liefert die ID des Ereignisses.
stEventEntry	TcEventEntry [▶ 113]	Get	Liefert die Ereignisdefinition.

## Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 4.2.1 EqualsTo



Diese Methode führt einen Vergleich mit einem am Eingang angegebenen anderen Ereignis aus.

#### Syntax

```

METHOD EqualsTo : BOOL
VAR_INPUT
    ipOther : I_TcEventBase;
END_VAR
  
```

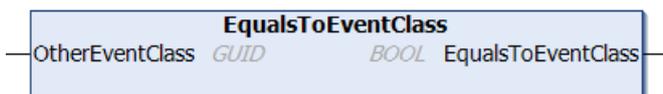
## Eingänge

Name	Typ	Beschreibung
ipOther	I_TcEventBase [▶ 105]	Zu vergleichendes Ereignis

## Rückgabewert

Name	Typ	Beschreibung
EqualsTo	BOOL	Liefert TRUE, wenn die Ereignisse übereinstimmen.

### 4.2.2 EqualsToEventClass



Diese Methode führt einen Vergleich mit einer am Eingang angegebenen anderen Ereignisklasse aus.

#### Syntax

```

METHOD EqualsToEventClass : BOOL
VAR_INPUT
    OtherEventClass : GUID;
END_VAR
  
```

 Eingänge

Name	Typ	Beschreibung
OtherEventClass	GUID	Zu vergleichende Ereignisklasse.

 Rückgabewert

Name	Typ	Beschreibung
EqualsToEventClass	BOOL	Liefert TRUE, wenn die Ereignisklassen übereinstimmen.

### 4.2.3 EqualsToEventEntry



Diese Methode führt einen Vergleich mit einem am Eingang angegebenen anderen Ereignis aus.

**Syntax**

```

METHOD EqualsToEventEntry : BOOL
VAR_INPUT
    OtherEventClass : GUID;
    nOtherEventID : UDINT;
    eOtherSeverity : TcEventSeverity;
END_VAR
    
```

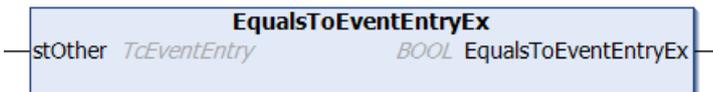
 Eingänge

Name	Typ	Beschreibung
OtherEventClass	GUID	Ereignisklasse des zu vergleichenden Ereignisses.
nOtherEventID	UDINT	Event-ID des zu vergleichenden Ereignisses.
eOtherSeverity	<u>TcEventSeverity</u> [ <a href="#">▶ 113</a> ]	Event-Severity des zu vergleichenden Ereignisses.

 Rückgabewert

Name	Typ	Beschreibung
EqualsToEventEntry	BOOL	Liefert TRUE, wenn die Ereignisse übereinstimmen.

### 4.2.4 EqualsToEventEntryEx



Diese Methode führt einen Vergleich mit einem am Eingang angegebenen anderen Ereignis aus.

**Syntax**

```

METHOD EqualsToEventEntryEx : BOOL
VAR_INPUT
    stOther : TcEventEntry;
END_VAR
    
```

### 📌 Eingänge

Name	Typ	Beschreibung
stOther	TcEventEntry [▶_113]	Zu vergleichendes Ereignis.

### 📌 Rückgabewert

Name	Typ	Beschreibung
EqualsToEventEntryEx	BOOL	Liefert TRUE, wenn die Ereignisse übereinstimmen.

## 4.2.5 GetJsonAttribute

GetJsonAttribute		
sJsonAttribute	REFERENCE TO STRING	HRESULT GetJsonAttribute
nJsonAttribute	UDINT	

Diese Methode liefert das JSON-Attribut.

### Syntax

```
METHOD GetJsonAttribute : HRESULT
VAR_INPUT
    sJsonAttribute : REFERENCE TO STRING;
    nJsonAttribute : UDINT;
END_VAR
```

### 📌 Eingänge

Name	Typ	Beschreibung
sJsonAttribute	REFERENCE TO STRING	Referenz auf eine Variable vom Typ String
nJsonAttribute	UDINT	Länge der String-Variable

### 📌 Rückgabewert

Name	Typ	Beschreibung
GetJsonAttribute	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Liefert ERROR_BAD_LENGTH, wenn die Länge der Variable zu klein ist. Ansonsten wird HRESULT als Fehlercode zurückgegeben.

## 4.2.6 RequestEventClassName

RequestEventClassName		
nLangId	DINT	BOOL RequestEventClassName
sResult	REFERENCE TO STRING	BOOL bError
nResultSize	UDINT	HRESULT hrErrorCode

Diese Methode liefert den Namen der Ereignisklasse.

### Syntax

```
METHOD RequestEventClassName : BOOL
VAR_INPUT
    nLangId      : DINT;
    sResult      : REFERENCE TO STRING;
    nResultSize  : UDINT;
END_VAR
VAR_OUTPUT
```

```
bError      : BOOL;
hrErrorCode : HRESULT;
END_VAR
```

**Eingänge**

Name	Typ	Beschreibung
nLangId	DINT	Spezifiziert die Sprach-ID Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ...
sResult	REFERENCE TO STRING	Referenz auf eine Variable vom Typ String
nResultSize	UDINT	Größe der String-Variable in Bytes

**Rückgabewert**

Name	Typ	Beschreibung
RequestEventClassName	BOOL	Liefert TRUE, sobald die Abfrage beendet wurde. Wenn die asynchrone Abfrage noch aktiv ist, liefert der Rückgabewert FALSE. Die Methode muss so lange aufgerufen werden, bis der Rückgabewert TRUE ist.

**Ausgänge**

Name	Typ	Beschreibung
bError	BOOL	Liefert FALSE, wenn der Methodenaufruf erfolgreich war. Liefert TRUE, wenn ein Fehler aufgetreten ist.
hrErrorCode	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Im Falle eines Fehlers wird ein Fehlercode ausgegeben.

### 4.2.7 RequestEventText



Diese Methode liefert den Ereignistext.

**Syntax**

```
METHOD RequestEventText : BOOL
VAR_INPUT
  nLangId      : DINT;
  sResult      : REFERENCE TO STRING;
  nResultSize  : UDINT;
END_VAR
VAR_OUTPUT
  bError       : BOOL;
  hrErrorCode  : HRESULT;
END_VAR
```

### 🚪 Eingänge

Name	Typ	Beschreibung
nLangId	DINT	Spezifiziert die Sprach-ID Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ...
sResult	REFERENCE TO STRING	Referenz auf eine Variable vom Typ String
nResultSize	UDINT	Größe der String-Variablen in Bytes

### 🚪 Rückgabewert

Name	Typ	Beschreibung
RequestEventText	BOOL	Liefert TRUE, sobald die Abfrage beendet wurde. Wenn die asynchrone Abfrage noch aktiv ist, liefert der Rückgabewert FALSE. Die Methode muss so lange aufgerufen werden, bis der Rückgabewert TRUE ist.

### 🚪 Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Liefert FALSE, wenn der Methodenaufruf erfolgreich war. Liefert TRUE, wenn ein Fehler aufgetreten ist.
hrErrorCode	HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war. Im Falle eines Fehlers wird ein Fehlercode ausgegeben.

## 4.3 I\_TcMessage

Diese Schnittstelle stellt Methoden und Eigenschaften für das Nachrichten-Handling bereit.

### Vererbungshierarchie

[I\\_TcEventBase](#) [▶ 105]

I\_TcMessage

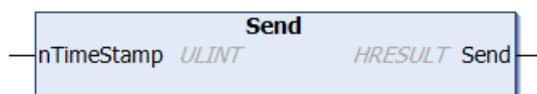
### 🔗 Methoden

Name	Beschreibung
<a href="#">Send</a> [▶ 110]	Sendet eine Nachricht

### Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 4.3.1 Send



Diese Methode sendet die Nachricht.

**Syntax**

```
METHOD Send : HRESULT
VAR_INPUT
    nTimeStamp: ULINT;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
nTimeStamp	ULINT	0: Aktueller Zeitstempel wird verwendet > 0: Externer Zeitstempel in 100 Nanosekunden seit dem 1. Januar 1601 (UTC).

 **Rückgabewert**

Name	Typ	Beschreibung
Send	FB_HRESULT	Liefert S_OK, wenn der Methodenaufruf erfolgreich war, ansonsten ein HRESULT als Fehlercode

## 4.4 I\_TcSourceInfo

Diese Schnittstelle definiert Eigenschaften für eine Quellinformation.

 **Methoden**

Name	Beschreibung
<a href="#">EqualsTo [▶ 111]</a>	Vergleicht eine Instanz mit Quellinformationen mit einer anderen Instanz.

 **Eigenschaften**

Name	Typ	Zugriff	Beschreibung
guid	GUID	Get	Liefert die GUID der Quelleinfo zurück.
nId	UDINT	Get	Liefert die ID der Quelleinfo zurück.
sName	STRING(ParameterList.cSourceNameSize-1)	Get	Liefert den Namen der Quelleinfo zurück.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4022.20	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger

### 4.4.1 EqualsTo



Diese Methode vergleicht eine Instanz mit Quellinformationen mit einer anderen Instanz.

**Syntax**

```
METHOD EqualsTo : BOOL
VAR_INPUT
    ipOther : I_TcSourceInfo;
END_VAR
```

 **Eingänge**

Name	Typ	Beschreibung
ipOther	<a href="#">I_TcSourceInfo [► 111]</a>	Zu vergleichende Quellinformationen

 **Rückgabewert**

Name	Typ	Beschreibung
EqualsTo	BOOL	Liefert TRUE, wenn die Quellinformationen übereinstimmen.

## 5 Datentypen

### 5.1 TcEventEntry

Definiert ein Ereignis (Event) mittels Ereignisklasse, Ereignis-ID und Severity.

#### Syntax

Definition:

```
TYPE TcEventEntry :
STRUCT
    uuidEventClass : GUID;
    nEventId       : UDINT;
    eSeverity      : TcEventSeverity;
END_STRUCT
END_TYPE
```

#### Parameter

Name	Typ	Beschreibung
uuidEventClass	GUID	GUID der Ereignisklasse.
nEventId	UDINT	ID des Ereignisses.
eSeverity	TcEventSeverity	Event-Severity definiert den Schweregrad des Ereignisses.

### 5.2 TcEventSeverity

Definiert die Severity des Ereignisses.

#### Syntax

Definition:

```
{attribute 'qualified_only'}
TYPE TcEventSeverity : (
    Verbose := 0,
    Info    := 1,
    Warning := 2,
    Error   := 3,
    Critical := 4);
END_TYPE
```

#### Parameter

	Name	Beschreibung
4	Critical	Kritisch
3	Error	Fehler
2	Warning	Warnung
1	Info	Information
0	Verbose	Erweiterte Ausgabe

### 5.3 TcEventConfirmationState

Definiert den Bestätigungszustand eines Alarms.

#### Syntax

Definition:

```
{attribute 'qualified_only'}
TYPE TcEventConfirmationState : (
  NotSupported := 0,
  NotRequired  := 1,
  WaitForConfirmation := 2,
  Confirmed    := 3,
  Reset        := 4);
END_TYPE
```

### Parameter

Name	Beschreibung
Confirmed	Bestätigt
NotRequired	Bestätigung im aktuellen Zustand nicht nötig. (Alarm aktuell nicht im Zustand Raised.)
NotSupported	Wurde ohne Bestätigung initialisiert.
Reset	Initialzustand
WaitForConfirmation	Wartet auf Bestätigung.

## 6 Globale Listen

### 6.1 Global\_Constants

```
VAR_GLOBAL CONSTANT
    EMPTY_EVENT_CLASS : GUID := (Data1:=16#0, Data2:=16#0, Data3:=16#0, Data4:=[16#0,16#0,16#0,16#0,
16#0,16#0,16#0,16#0]);
    EMPTY_EVENT_ID : UDINT := 16#0;
    EMPTY_SEVERITY : TcEventSeverity := TcEventSeverity.Verbose;
    SUCCESS_EVENT : TcEventEntry := ( uuidEventClass := EMPTY_EVENT_CLASS, nEventID := EMPTY_EVENT_ID, eSeverity := EMPTY_SEVERITY );
END_VAR
```

Name	Typ	Initialwert
EMPTY_EVENT_CLASS	GUID	STRUCT(Data1:=16#0, Data2:=16#0, Data3:=16#0, Data4:=[16#0,16#0,16#0,16#0,16#0,16#0,16#0,16#0])
EMPTY_EVENT_ID	UDINT	16#0
EMPTY_SEVERITY	<a href="#">TcEventSeverity</a> ▶ <a href="#">113</a>	TcEventSeverity.Verbose
SUCCESS_EVENT	<a href="#">TcEventEntry</a> ▶ <a href="#">113</a>	STRUCT(uuidEventClass := EMPTY_EVENT_CLASS, nEventID := EMPTY_EVENT_ID, eSeverity := EMPTY_SEVERITY)

### 6.2 GVL

```
{attribute 'qualified_only'}
VAR_GLOBAL
    nLangId_OnlineMonitoring : DINT := 1033;
END_VAR
```

Name	Typ	Initialwert	Beschreibung
nLangId_OnlineMonitoring	DINT	1033	Sprachkennung für das Online Monitoring Englisch (en-US) = 1033 Deutsch (de-DE) = 1031 ...

### 6.3 Parameterlist

```
{attribute 'qualified_only'}
VAR_GLOBAL CONSTANT
    cSourceNameSize : UDINT(81..10000) := 256;
END_VAR
```

Name	Typ	Initialwert	Beschreibung
cSourceNameSize	UDINT(81..10000)	256	Größe in Bytes für den Namen der Quellinformation. Empfohlen sind maximal 512 Bytes.

### 6.4 Global\_Version

Alle Bibliotheken haben eine bestimmte Version. Diese Version ist u. a. im SPS-Bibliotheksrepository zu sehen.

Eine globale Konstante enthält die Information über die Bibliotheksversion (vom Typ ST\_LibVersion):

Global\_Version

```
VAR_GLOBAL CONSTANT
    stLibVersion_Tc3_EventLogger : ST_LibVersion;
END_VAR
```

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).

## 7 Beispiele

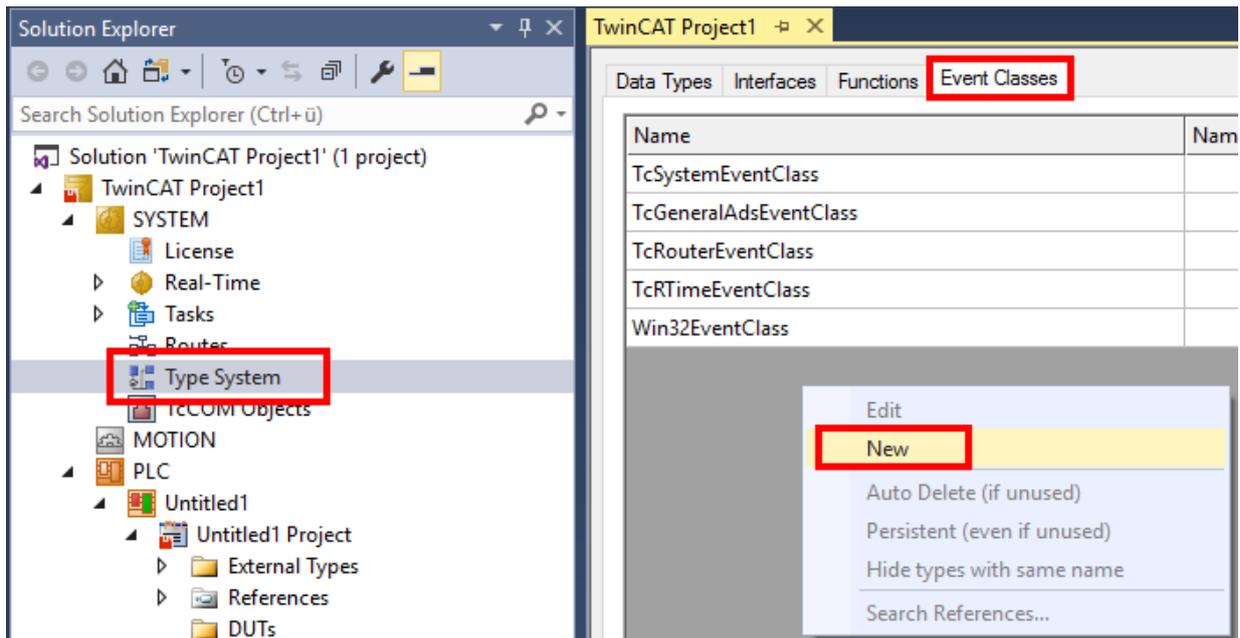
### 7.1 Tutorial

Dieses Tutorial verdeutlicht die Arbeitsschritte von einem leeren TwinCAT-Projekt bis hin zu einer abgeschickten Meldung. Es zeigt die im Abschnitt Technische Einführung beschriebenen Eigenschaften des TwinCAT 3 EventLoggers anschaulich im Arbeitsablauf.

#### Ereignisklasse im Typsystem von TwinCAT anlegen

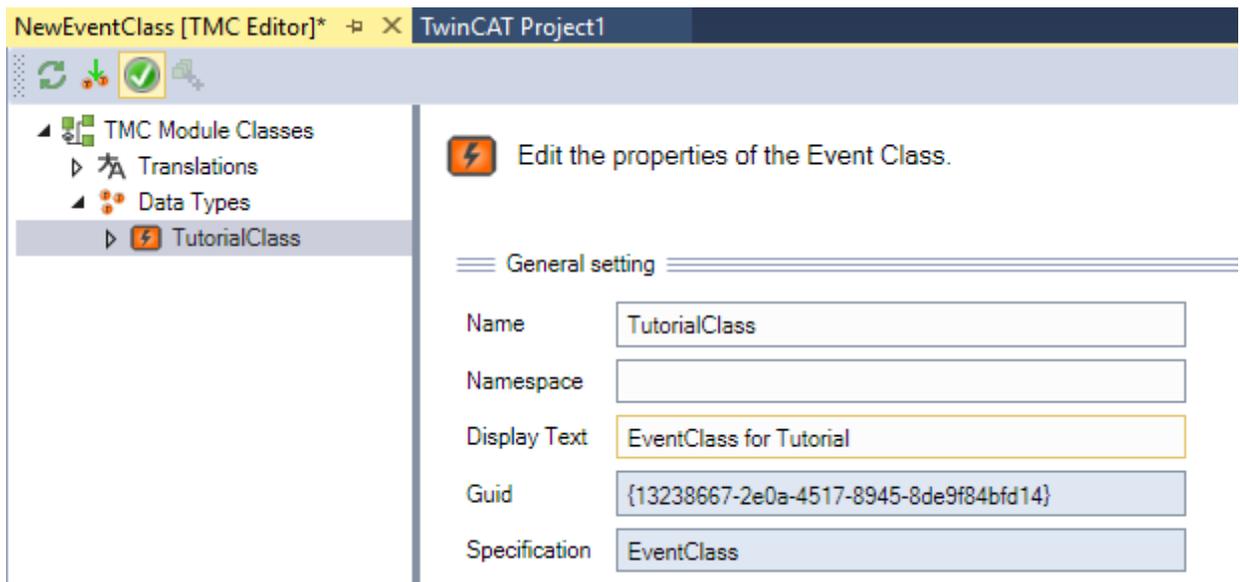
✓ Ein Standard-TwinCAT-SPS-Projekt existiert.

1. Klicken Sie im SYSTEM-Teilbaum doppelt auf **Type System** und wählen Sie in dem sich öffnenden Editor die Registerkarte **Event Classes**. Öffnen Sie das Kontextmenü und wählen Sie den Befehl **New**.

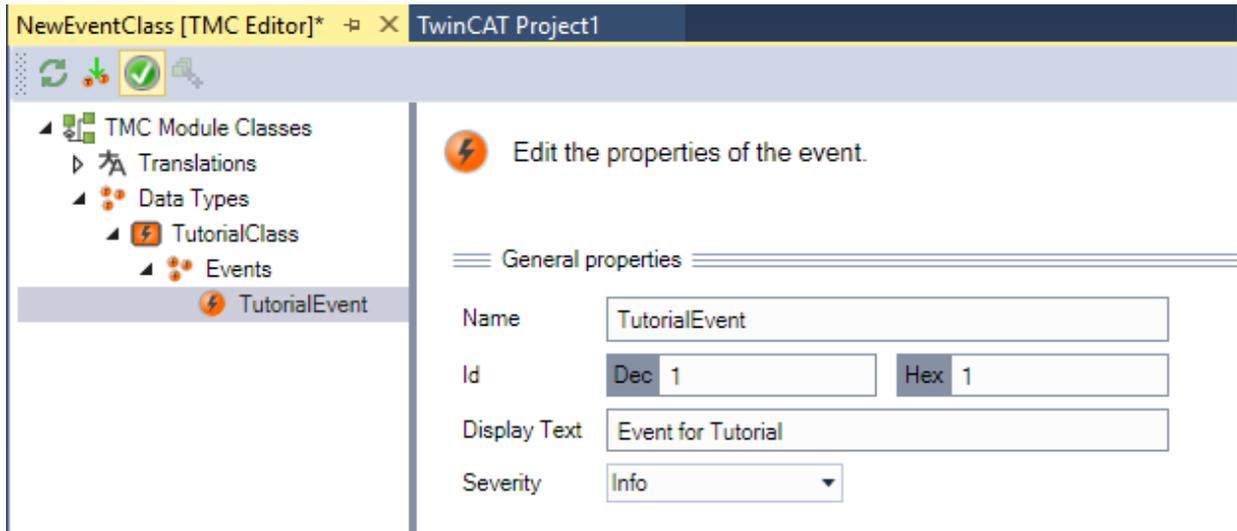


⇒ Der TMC Editor öffnet sich.

2. Geben Sie der Ereignisklasse einen Namen und geben Sie einen Display-Text an.



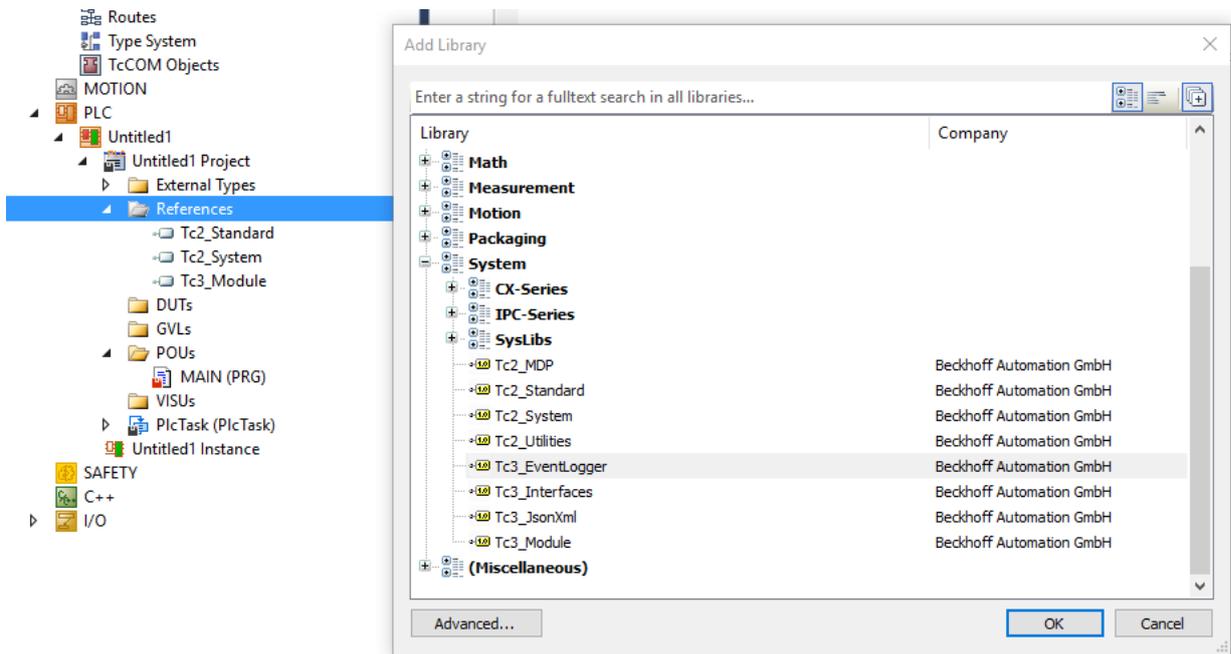
- Unterhalb der Ereignisklasse ist bereits ein Ereignis angelegt. Geben Sie dem Ereignis einen Namen und geben Sie einen Display-Text und die Severity an.



- Speichern und ggf. schließen Sie die Ereignisklasse.
  - ⇒ Der Quellcode wird in der SPS bereitgestellt und ist unter dem Symbol TC\_EVENTS erreichbar.

**Bibliothek TC3\_EventLogger hinzufügen**

- Wählen Sie im Kontextmenü des Objekts **References** den Befehl **Add Library**.
  - ⇒ Der Dialog **Add Library** öffnet sich.
- Wählen Sie die Bibliothek aus und bestätigen Sie den Dialog.



⇒ Die Bibliothek wird dem SPS-Projekt hinzugefügt.

**SPS-Programm erstellen**

- Öffnen Sie mit einem Doppelklick das MAIN-Programm des SPS-Projekts im Editor.
- Deklariert und initialisieren Sie die Variablen bInit und bSend und deklarieren Sie eine Instanz des Funktionsbausteins FB\_TcMessage:

```
PROGRAM MAIN
VAR
    bInit : BOOL := TRUE;
    bSend : BOOL := TRUE;
```

```

    fbMsg : FB_TcMessage;
END_VAR

```

3. Implementieren Sie den Sendevorgang wie im Code dargestellt. Die Nachricht wird einmalig mittels der CreateEx-Methode initialisiert. Da die Initialisierung dynamische Ressourcen benötigt, sollte sie nicht zyklisch erfolgen. Die initialisierte Nachricht wird anschließend mit der Send-Methode gesendet.

```

IF bInit THEN
    bInit := FALSE;
    fbMsg.CreateEx(TC_EVENTS.TutorialClass.TutorialEvent, 0);
END_IF

IF bSend THEN
    bSend := FALSE;
    fbMsg.Send(0);
END_IF

```

4. Erstellen Sie das SPS-Projekt und starten Sie die SPS.

⇒ Das Ergebnis wird im Fenster LoggedEvents des TwinCAT 3 Engineerings angezeigt.

Severity Level	EventClassName	EventId	Text	SourceName	SourceId	Time Raised
Info	EventClass for Tutorial	1	Event for Tutorial	MAIN	0x08502000	19.05.2018 14:25:54.225

## 7.2 Beispiel ResultMessage

Dieses Beispiel zeigt die Verwendung des TwinCAT 3 Eventloggers mit Funktionsbausteinen. Es demonstriert zum einen, wie ein Ausgang an einem Funktionsbaustein dazu verwendet werden kann, die Event-Informationen als erweiterte Rückgabe zu nutzen. Zum anderen demonstriert es, wie eine Parametrisierung vorgenommen werden kann, um eine Ausgabe der Meldungen über den TwinCAT 3 Eventlogger nur in bestimmten Fällen durchzuführen.

Download des Beispiels: [https://github.com/Beckhoff/Tc3Eventlogger\\_Samples/tree/main/PLC/Tc3EventLogger\\_ResultMessageSample](https://github.com/Beckhoff/Tc3Eventlogger_Samples/tree/main/PLC/Tc3EventLogger_ResultMessageSample)

Das Beispiel besteht aus zwei Funktionsbausteinen:

- **FB\_MathCalculation:** Dieser Funktionsbaustein bietet zwei Methoden und zwei Properties an, die Meldungen immer am Ausgang ipResultMessage ausgeben und zusätzlich über den EventLogger absenden, wenn ein Tracelevel überschritten ist.
  - Methode Addition(): Addiert zwei Zahlen und sendet bei einem Überlauf eine Nachricht
  - Methode Divison(): Dividiert zwei Zahlen nach Prüfung. Sendet eine Nachricht, wenn eine Division durch 0 erfolgt.
  - Property bTraceLevelDefault: Gibt an, ob das Tracelevel lokal am Funktionsbaustein beachtet werden soll, oder ob ein Library Tracelevel verwendet werden soll, welcher im Beispiel in der GVL vorhanden ist.
  - Property eTraceLevel: Die Methoden senden die Nachricht nur über den EventLogger ab, wenn die Severity größer oder gleich diesem Property ist.
- **FB\_Control:** Dieser Funktionsbaustein zeigt die Verwendung des FB\_MathCalculation-Bausteins innerhalb eines anderen Bausteins. Dabei nutzt die Execute-Methode des FB\_Control die FB\_MathCalculation.Divison() und behandelt die Nachricht als Fehlercode selbst weiter.

## 7.3 Beispiel Listener

Dieses Beispiel zeigt die Verwendung des TwinCAT 3 Eventloggers in Bezug auf Nachrichten und Alarme. Gleichzeitig wird das Empfangen von Nachrichten in einem zweiten Projekt gezeigt.

Download des Beispiels: [https://github.com/Beckhoff/Tc3Eventlogger\\_Samples/tree/main/PLC/Tc3EventLogger\\_ListenerSample](https://github.com/Beckhoff/Tc3Eventlogger_Samples/tree/main/PLC/Tc3EventLogger_ListenerSample)

**Publisher-Projekt**

Im Publisher-Projekt werden einfache BOOL-Variablen als Trigger verwendet:

- bSendMessage, um eine Nachricht abzusetzen.
- bRaiseAlarm, um einen Alarm zu setzen.
- bClearAlarm, um einen Alarm zurückzunehmen.
- bConfirmAlarm, um einen Alarm zu quittieren.

Zusätzlich gibt es die Möglichkeit, das JSON-Attribut zu setzen, um dieses bei beiden Nachrichten mitzusenden.

**Listener-Projekt**

Im Listener-Projekt ist ein Funktionsbaustein FB\_Listener enthalten, der den in der Tc3\_EventLogger enthaltenen Baustein FB\_ListenerBase erweitert. Der Baustein implementiert hierbei die Funktionen zum Empfang der Nachrichten:

- OnMessageSent: Wenn eine Nachricht versendet wurde, wird der EventLogger diese Methode als Callback aufrufen. Die Methode zählt die Anzahl der Nachrichten mit.
- OnAlarmRaised/OnAlarmCleared/OnAlarmConfirmed: Wenn der Alarm den Zustand ändert, wird der EventLogger diese Methode als Callback aufrufen. Die Methoden zählen jeweils die Anzahl der Zustandsänderungen mit.
- Um den Empfang der Nachrichten zu initiieren, ist eine Execute-Methode an dem Baustein implementiert.
- Der Text der letzten empfangenen Nachricht kann abgeholt werden.
- Der Funktionsbaustein FB\_ListenerTest nutzt den FB\_Listener. Hierbei registriert er einmalig die zu empfangende Ereignisklasse. Eine weitere existierende Ereignisklasse wird nicht empfangen, wodurch die Filterfunktionalität demonstriert wird.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.17	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>= v3.1.27.0)

**7.4 Beispiel Filter**

Dieses Beispiel zeigt die Verwendung des TwinCAT 3 EventLoggers in Bezug auf das Empfangen von Nachrichten. Hierbei wird ein Fokus auf die Filterfunktionen gesetzt, um zielgerichtet die richtigen Nachrichten zu verarbeiten.

Download des Beispiels: [https://github.com/Beckhoff/Tc3Eventlogger\\_Samples/tree/main/PLC/Tc3EventLogger\\_FilterSample](https://github.com/Beckhoff/Tc3Eventlogger_Samples/tree/main/PLC/Tc3EventLogger_FilterSample)

Das Beispiel besteht aus vier Komponenten:

- Es werden eine Reihe von unterschiedlichen Nachrichten abgesendet, wodurch die Selektion der Nachrichten in unterschiedlichen Filtern demonstriert wird.
- Eine Komponente zeigt, wie aus dem Cache Nachrichten verworfen werden können, welche über einen Filter spezifiziert werden.
- Eine andere Komponente zeigt den Export von im Cache hinterlegten Nachrichten in eine CSV-Datei. Auch hierbei wird über die Filter programmiert, welche Nachrichten ausgewählt werden sollen.
- Eine weitere Komponente zeigt das allgemeine Empfangen von in der Echtzeit gesendeten Nachrichten sowie das Empfangen von EtherCAT Emergency Nachrichten, welche empfangen werden sollen.

**Voraussetzungen**

Entwicklungsumgebung	Zielplattform	Einzubindende SPS-Bibliotheken
TwinCAT v3.1.4024.17	PC oder CX (x64, x86, Arm®)	Tc3_EventLogger (>= v3.1.27.0)

## **Trademark statements**

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.

## **Third-party trademark statements**

Arm, Arm9 and Cortex are trademarks or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere.

Mehr Informationen:  
**[www.beckhoff.com/te1000](http://www.beckhoff.com/te1000)**

Beckhoff Automation GmbH & Co. KG  
Hülshorstweg 20  
33415 Verl  
Deutschland  
Telefon: +49 5246 9630  
[info@beckhoff.com](mailto:info@beckhoff.com)  
[www.beckhoff.com](http://www.beckhoff.com)

